ThinkCore DA-660-8/16-CE
User's Manual

Second Edition, May 2006

www.moxa.com/product

Moxa Systems Co., Ltd.
Tel: +886-2-8919-1711
Fax:+886-2-8919-1722
Web: www.moxa.com

MOXA Technical Support
Worldwide: support@moxa.com
The Americas support@usa.moxa.com

ThinkCore DA-660-8/16-CE User's Manual

The software described in this manual is furnished under a license agreement and may be used only in accordance with the terms of that agreement.

Copyright Notice

Copyright © 2006 Moxa Systems Co., Ltd.
All rights reserved.
Reproduction without permission is prohibited.

Trademarks

MOXA is a registered trademark of The Moxa Group.
All other trademarks or registered marks in this manual belong to their respective manufacturers.

Disclaimer

Information in this document is subject to change without notice and does not represent a commitment on the part of Moxa.

Moxa provides this document "as is," without warranty of any kind, either expressed or implied, including, but not limited to, its particular purpose. Moxa reserves the right to make improvements and/or changes to this manual, or to the products and/or the programs described in this manual, at any time.

Information provided in this manual is intended to be accurate and reliable. However, Moxa Technologies assumes no responsibility for its use, or for any infringements on the rights of third parties that may result from its use.

This product might include unintentional technical or typographical errors. Changes are periodically made to the information herein to correct such errors, and these changes are incorporated into new editions of the publication.

Table of Contents

Chapter 1 Introduction 1-1

Overview 1-2
- Package Checklist 1-2
- Model Descriptions and Features 1-2
- Hardware Specifications 1-2

Software Features 1-3
- Applications Development Environment 1-3
- Networking and Communications Capabilities 1-4
- Supported Servers and Daemons 1-4

Obtaining the Firmware Build Version 1-5
Memory and File Systems 1-5
Hive-Based Registry 1-5
RS-232/422/485 Serial Ports 1-5

Chapter 2 Getting Started 2-1

Starting Your DA-660 Computer 2-2
Resetting Your DA-660 Computer 2-2
Operating Your DA-660 Computer Via Serial Console 2-2
Changing the Network Settings 2-2
Operating Your DA-660 Computer Via Telnet Client 2-3
User/Group Management 2-3
Adjusting System Time and RTC Time 2-4
Starting and Stopping Services 2-5
Troubleshooting Network Connectivity 2-5
Simple Network Management Protocol (SNMP) 2-6
- SNMP Agent 2-6
- SNMP Manager 2-7

Chapter 3 Web-based Management System 3-1

System Information 3-2
Networking/Server Configuration 3-2
Serial Port Configuration 3-3
Process (Thread) Monitoring/Control 3-3
Launching Processes Automatically 3-4
Services Monitoring/Control 3-4
Binary/Text File Management 3-5

Chapter 4 Application Development 4-1

Installing Development Tools (IDEs) 4-2
- VB.NET/C# Applications: Using Visual Studio 2005 4-2
- C/C++ Applications: Using eMbedded Visual C++ (eVC) 4.0 4-2

Installing MOXA Windows® CE 5.0 C/C++ SDKs 4-3
Developing Applications with IDEs 4-6
- Developing eMbedded Visual C++ 4.0 applications with an imported SDK 4-7
- Developing VB.NET / C# applications with a .NET Compact Framework 4-8

Visual Studio 2005 Remote Tools 4-9
- Using Visual Studio C#, VB.Net or C++ 2005 Express 4-9
- Remote Tool # 1: Application Debugger 4-11
- Remote Tool #2: Remote File Explorer 4-15

Chapter 5 Programming Examples 5-1
Embedded Visual C++4 Examples 5-1
Example #1—MOXA UART Supporting RS-232/422/485 5-2
Example #2—Buzzer 5-5
Visual Studio 2005 C# Examples 5-5
Example #1—MOXA UART Supports RS-232/422/485 5-6
Example #2—Buzzer 5-9
Example #3—Real Time Clock 5-10
Frequently Asked Questions 5-10
Appendix A Using the LCM Keypad A-0
Appendix B Firmware Upgrade Procedure B-0
Appendix C Service Information C-1
MOXA Internet Services C-2
Problem Report Form C-3
Product Return Procedure C-4

1

Introduction

Microsoft® Windows® CE 5.0 is an open, scalable, 32-bit operating system (OS) that allows you to build a wide range of innovative, small footprint devices. A typical Windows® CE-based device is designed for a specific use, and often runs disconnected from other computers, or distributed as a front-end to a centralized host. Examples include enterprise tools, such as industrial controllers, communications hubs, and point-of-sale terminals, and display devices, such as HMI, advertisement appliances, and interactive panels.

The ThinkCore DA-660 WinCE series of ready-to-run embedded computers includes the 8-port DA-660-8-CE and the 16-port DA-660-16-CE.

The following topics are covered in this chapter:

- **Overview**
  - Package Checklist
  - Model Descriptions and Product Features
  - Product Hardware Specifications
- **Software Features**
  - Applications Development Environment
  - Networking and Communications Capabilities
  - Supported Servers and Daemons
- **Obtaining the Firmware Build Version**
- **Memory and File Systems**
- **Hive-Based Registry**
- **RS-232/422/485 Serial Ports**

Overview

The ThinkCore DA-660 WinCE embedded computer is ideal for reliable, small-footprint, programmable, communication-centric applications. DA-660 uses an Intel XScale IXP-422 RISC-based CPU running at a clock rate of 266 MHz. Unlike the CISC architecture under an X86 CPU, the RISC design supports compact instruction sets. A RISC-based CPU generates less heat and consequently enhances reliability. To provide sufficient data storage and run-time memory for your applications, DA-660 integrates 32 MB of Flash memory and 128 MB of SDRAM. In addition, the DA-660 has two LAN ports and 8 or 16 serial ports to extend its communication capabilities. Two efficient LAN ports support high-performance data transmission over the Intranet/Internet. Each serial port supports an RS-232/422/485 3-in-1 operation mode switch to provide centrally-controlled connectivity to diverse serial devices.

The Da-660 WinCE system is built on the Microsoft® Windows® CE 5.0 operating system (OS). Using a common, widely used programming environment makes DA-660 suitable for new system development and legacy system migration.

The hardware and software capabilities of the Da-660 WinCE system make it an ideal computing platform for data acquisition and transmission applications.

Package Checklist

Before installing the Da-660 WinCE system, verify that the package contains the following items.

- 1 ThinkCore DA-660 WinCE
- 19-inch Rackmount Kit
- ThinkCore DA-660 WinCE Quick Installation Guide
- ThinkCore DA-660 WinCE Document & Software CD
- Cross-over Ethernet cable
- CBL-RJ45M9-150: 150 cm, 8-pin RJ45 to male DB9 serial port cable
- CBL-RJ45F9-150: 150 cm, 8-pin RJ45 to female DB9 console port cable
- Power Cord
- Product Warranty Sheet

Notify your sales representative if any of above items is missing or damaged.

Model Descriptions and Features

The Da-660 WinCE Series (referred to in this manual as DA-660) consists of two models. Both models have exactly the same software features and hardware specifications, but DA-660-8-CE has 8 serial ports and DA-660-16-CE has 16 serial ports.

The following table lists the hardware specifications of the models in more detail.

Hardware Specifications

Model		DA-660-8-CE	DA-660-16-CE
CPU		Intel XScale IXP-422 266 MHz	
RAM		128 MB	
Flash		32 MB	
LAN		Auto-sensing 10/100 Mbps × 2 with built-in 1.5 KV magnetic isolation protection, RJ45 Connector	
Serial Port		RS-232/422/485 × 8 RJ45 Connector	RS-232/422/485 × 16 RJ45 Connector
	Serial Protection	15 KV ESD	
	Data Bits	5, 6, 7, 8	
	Stop Bits	1, 1.5, 2	
	Parity	None, Even, Odd, Space, Mark	
	Flow Control	RTS/CTS, XON/XOFF, RS-485 ADDC™	
	Speed	50 bps to 921.6 Kbps	
Serial Console		RS-232 × 1, RJ45 Connector	
LCM		128 × 32 dots	
Keypad		4 buttons	
Real Time Clock		Yes	
Buzzer		Yes	
Reset Button		Reset to Default × 1	
Power Input		100- to 240 VAC/VDC auto ranging (47 to 63 Hz for AC input)	
Power Consumption		12W	
Dimensions (W × D × H)		480 × 198 × 45 mm (with rack-mount ears) 440 × 198 × 45 mm (without rack-mount ears)	
Operating temperature		-10 to 60°C (14 to 140°F), 5 to 95% RH	
Storage temperature		-20 to 80°C (-4 to 176°F), 5 to 95% RH	
Regulatory Approvals		EMC: CE Class A, FCC Class A Safety: UL, CUL, TUV	
Warranty		5 years	

Software Features

DA-660 ready-to-run embedded computers are network-centric, head-less computers that are designed to be programmable for embedded communication applications.

Applications Development Environment

The Windows® CE operating system provides an easy-to-use programming environment for the DA-660. In fact WinCE's popular application development features make programming as convenient and easy as in a PC environment.

- **C Libraries and Run-times**—Compared to the C libraries and run-times used on a desktop PC running Windows®, the C libraries and run-times on a DA-660 WinCE are a subset of the WIN32 APIs. The system supports a full ANSI C run-time, standard input/output library, standard input/output ASCII library, and standard ASCII string functions. In addition, compiler C++ exception handling and Run-Time Type Information (RTTI) equivalent to desktop C++ compilers are supported.
- **Component Services (COM and DCOM)**—The Common Object Model (COM) is an operating system-independent, object-oriented system for creating binary software components that can interact with other COM-based components in the same process space, in other processes, or on remote machines.

- **Microsoft® Foundation Classes (MFC)**—MFC is a comprehensive class library and complete object-oriented application framework designed to help build applications, COM components, and controls.
- **SOAP Toolkit**—SOAP is an XML-based protocol for object exchange and remote procedure calls. Microsoft® Windows® CE 5.0 provides functionality similar to the SOAP Toolkit version 2 on the desktop. It provides a layer that allows COM objects to use SOAP as the transport protocol for remote procedure calls and to interact with Web services.
- **Microsoft® .NET Compact Framework 2.0**—Offers a choice of programming languages, initially Microsoft® Visual Basic® and Microsoft® Visual C#, and eliminates the common problems faced with language interoperability.
- **XML**—Provides the Document Object Model (DOM) for base XML functionality, support for XML Query Language (XQL) and XPATH, Extensible Style Sheet Language Transformations (XSLT) that enables you to transform one class of XML document to another, SAX2 support for event-based parsing of XML documents (including MSXML Writer), and parsing based on Simple API for XML (SAX) for resource-constrained target devices.
- **Winsock 2.2**—Provides enhanced capabilities over Winsock 1.1, including installable service providers for additional third-party protocols, and Media sense.

Networking and Communications Capabilities

For network centric embedded application usage, DA-660 not only provides powerful communication hardware interfaces, including dual Ethernet and 3-in-1 serial ports, but also supports the networking and communications capabilities that are built into the Windows® CE 5.0 OS. The following features are supported:

- **Simple Network Management Protocol (SNMP)**—Monitors remote connections to the network.
- **Simple Network Time Protocol (SNTP) Client**—Provides support for synchronizing the device's system time with an SNTP server, and supports Daylight Savings Time.
- **Serial Communications**—In addition to the 16550 UART driver bound to a debug port and the console port, a special driver for 8 or 16 additional MOXA home-made serial ports is also included.
- **Network Utilities (IpConfig, Ping, Route)**—Utilities for troubleshooting various network problems.
- **TCP/IP**—Includes IP, Address Resolution (ARP), Internet Control Message (ICMP), Internet Group Membership (IGMP), Transmission Control (TCP), User Datagram (UDP), name resolution and registration, and DHCP.

Supported Servers and Daemons

In addition to the development and communication capabilities mentioned above, DA-660 embeds the services and daemons listed below. These common and easy-to-use application servers help users migrate the DA-660 embedded computer to industrial communication applications easily and conveniently.

- **Telnet Server**—A sample server that allows remote administration through a standard Telnet client.
- **FTP Server**—A sample server used for transferring files to and from remote computer systems over a network using TCP/IP.
- **Web Server (HTTPD)**—Includes ASP, ISAPI Secure Socket Layer support, SSL 2, SSL 3, Transport Layer Security (TLS/SSL 3.1) public key-based protocols, and Web Administration ISAPI Extensions.
- **Dial-up Networking**—Consists of a RAS client API and Point to Point Protocol (PPP). RAS and PPP support Extensible Authentication Protocol (EAP) and RAS scripting.

Obtaining the Firmware Build Version

There are three ways to obtain the firmware version of the DA-660 embedded computer. This information is particularly important for feature identification.

- Examine the welcome message after you log on to the computer.
- Log on to the Web-based management system (described in a later chapter) to view the system information.
- Check the firmware version on the LCM display.

Memory and File Systems

The operating system and user applications occupy about 112 MB of the 128 MB of SDRAM. The kernel image occupies the rest of the space.

DA-660's internal file system controls access to the ROM, and also provides file storage in the object store, which is in the RAM. The ROM file system provides persistent storage for applications and related data, even when the main power supply is lost. The ROM file system integrates the read-only files that are stored in Flash ROM with the read/write files of both an application and a user. A child directory named "NORFlash" under the root directory indicates the ROM storage of the flash memory of size 15.5 MB.

The root directory is a RAM file system of size 15.5 MB. The root directory can be used for storing temporary files for your applications. Do not place persistent files or applications in the root directory because they will be deleted when the system is shut down. Instead, place them in the directory "NORFlash."

The internal file system supports TFAT (Transaction-Safe File Allocation Table), which is safer than FAT. TFAT protects the file system from write corruption in critical times, such as power loss. During power loss, the file state is rolled back. In addition, TFAT is a superset of FAT. This means that the internal file system of the embedded computer provides the ability to mount external FAT file systems.

Hive-Based Registry

The registry for DA-660 is a hive-based registry instead of a RAM-based registry. The hive-based registry stores registry data inside files, or hives, which can be kept on any file system. This eliminates the need for performing backup and restore on power off.

RS-232/422/485 Serial Ports

The DA-660 embedded computer comes with two embedded serial ports, named COM1 and COM2. COM1 is hidden inside the box case for debugging purposes when developing firmware. COM2 is used as the console port. Although the console port can be used when developing applications (e.g., data acquisition and control), we suggest that you always use the serial ports located on the front of the device for your applications, and reserve COM2 for console access.

The additional serial ports are named COM3, COM4, etc., reading from left to right. The ports are designed for reliable, high-speed data transmission. The ports are 3-in-1 (i.e., they can be set for RS-232, RS-422, or RS-485) to make them suitable for many different applications. Each of the ports supports a baudrate up to 921600 bps.

2

Getting Started

In this chapter, we explain how to use a PC to operate a DA-660 computer. For clarity, the PC is referred to as a development workstation and the DA-660 computer is called a target computer.

In addition, manual steps are described to facilitate operations such as system time adjustment, troubleshooting network connectivity, etc. Some of these operations can be done via system commands after gaining access to the computer and others can be done by a Web-based management system, which is described in a later chapter.

The following topics are covered in this chapter:

- **Starting Your DA-660 Computer**
- **Resetting Your DA-660 Computer**
- **Operating Your DA-660 Computer Via Serial Console**
- **Changing the Network Settings**
- **Operating Your DA-660 Computer Via Telnet Client**
- **User/Group Management**
- **Adjusting System Time and RTC Time**
- **Starting and Stopping Services**
- **Troubleshooting Network Connectivity**
- **Simple Network Management Protocol (SNMP)**
  - SNMP Agent
  - SNMP Manager

Starting Your DA-660 Computer

Connect the power cord that is shipped with the package to the AC/DC power connector (located on the right side of the rear panel) of the target computer. It takes about 30 to 60 seconds for the system to boot up. Once the system is ready, the **Ready** LED lights up and glows steadily till the computer is shutdown and the computer displays its model name and firmware version on the LCM.

Resetting Your DA-660 Computer

Cold-Start: Switch off and then switch on the power again. The computer reboots itself right away.

Resetting to Factory Defaults: To reset the system back to factory default settings, press and hold the **Reset to Default** button for at least 5 seconds. The buzzer sounds while the factory default settings are loaded. After the factory default settings have been loaded, the computer reboots itself.

Operating Your DA-660 Computer Via Serial Console

The serial console port (next to two LAN ports) gives users a convenient way of connecting the development workstation to the console utility of the target computer. This method is particularly useful when using the computer for the first time.

After you have wired a serial cable, go back to the development workstation and start a terminal program (e.g., HyperTerminal) by using the settings shown below for the serial console port.

Baudrate	115200 bps
Parity	None
Data bits	8
Stop bits	1
Flow Control	None
Terminal	ANSI

After a successful connection, type the login name and password as requested to logon the computer. The default values are both **"admin"**.

Login: admin

Password: admin

Changing the Network Settings

The DA-660 computer comes with two network interfaces. The default IP addresses and netmasks of the network interfaces are as follows:

	Default IP Address	**Netmask**
LAN 1	192.168.3.127	255.255.255.0
LAN 2	192.168.4.127	255.255.255.0

Normally, you are required to change them because they are located on a different local network from that of your development workstation. Use the command ***netconfig*** to complete the task. Type ***netconfig -h*** to examine the usage of the command.

```
\> netconfig -h

Usage: netconfig –n <"LAN1" or "LAN2"> [-m <netmask>] [-d <DNS server>] [-g <gateway>]
[-i <IP address>]
```

For example, your development workstation has a LAN port at 192.168.1.x and the Domain Name

Server (DNS) is at 192.168.2.6. Execute the following command.

```
\> netconfig –n LAN1 –i 192.168.1.5 –m 255.255.255.0 –g 192.168.1.254 –d 192.168.2.6
```

Use command ***netconfig*** to view the new settings.

```
\> netconfig
LAN1 Interface Configuration:
IP Address:   192.168.1.5
SubNet Mask:  255.255.255.0
Gateway:    192.168.1.254
DNS:      192.168.2.6

LAN2 Interface Configuration:
IP Address:   192.168.4.127
SubNet Mask:  255.255.255.0
Gateway:
DNS:
```

Operating Your DA-660 Computer Via Telnet Client

Before operating your target computer using the Telnet client, we suggest that you change the network settings of the computer (see the earlier section) to have at least one of the two network ports situated in the same LAN as your development workstation.

Use a cross-over Ethernet cable to connect your development workstation directly to the target computer, or a straight-through Ethernet cable to connect the computer to a LAN hub or switch. Next, use a Telnet client in your development workstation to make a connection to the Telnet console utility of the target computer. Upon a successful connection, type the login name and password as requested to log on to the computer.

After passing the authentication validity check, a group of commands will be available to operate the computer. Use ***HELP*** to display all of the commands, or type ***HELP [command name]*** to display extended help for the selected command. Some of these commands, such as ***DATE*** and ***TIME***, are very useful for managing the system time of the computer. Other commands, such as ***DIR*** and ***MKDIR***, are good utilities for file management. For example, to inspect the file structure of the root directory, type ***DIR***

```
\> dir /b
NORFlash
My Documents
Program Files
Temp
Windows
```

User/Group Management

User Group: You often want the target computer to be accessible by multiple users with different needs. For example, a specific service, such as ftp or telnet, is accessible only by the users within the permissible group. By default, three user groups, namely **ftpd**, **telnetd**, and **administrators** are pre-created for your convenience.

Adding a Group: Use the command ***useradd –g <groupName>*** to create a user group.

```
\> useradd –g yyyy
group yyyy has been added.
```

Deleting a Group: To remove a group, use the command ***userdel –g*** <***groupName***>.

```
\> userdel –g yyyy
group yyyy has been removed.
```

Adding a User: Use the command ***useradd*** <***newUserID***> to add a user for accessing the system. The user's password, by default, is the same as the user name.

```
\> useradd xxxx
user xxxx has been added.
```

In addition, you can permit this user to access a particular service by typing ***-g*** followed by the user group name of the service, i.e., ***useradd –g*** <***groupName***> <***newUserID***>. For example,

```
\> useradd –g telnetd xxxx
user xxxx is existent
group telnetd is existent
user xxxx has been added to group yyyy
```

Deleting a User: Use the command ***userdel*** <***userID***> to delete a user from the system. User "admin" **CANNOT** be deleted.

```
\> userdel xxxx
user xxxx has been deleted
```

You can also just remove a user from a user group by using the command ***userdel –g*** <***groupName***> <***newUserID***>. For example,

```
\> userdel –g yyyy xxxx
user xxxx has been removed from group yyyy
```

Changing the Password: Use the command ***"passwd*** <***userID***>***"*** to change your login password. The user's password, by default, is the same as the user name.

```
\> passwd xxxx
Current password:
New password:
Retype new password:
Password has been changed
```

Adjusting System Time and RTC Time

The DA-660 computer has two time settings, the system time (or CPU clock) and the RTC (Real Time Clock) time. The system time regulates the execution of instructions and the RTC keeps track of the time even when the computer is turned off. RTC time runs on a special battery that is not connected to the normal power supply. Do not confuse a computer's real-time clock with its CPU clock.

Setting the System Time Manually: Use the ***date***, and ***time*** commands to query the current system date/time or to set a new system date/time.

```
\> date
The current date is: Tuesday, November 22, 2005
Enter the new date (mm-dd-[yy]yy): 12-23-05

\> date /T
Wednesday, November 23, 2005
```

```
\> time
The current time is: 5:27:17 PM
Enter the new time (hh:mm:ss): 16:02:00

\> time /T
4:02:04 PM
```

Adjusting RTC Time: Use the command ***hwclock -w*** to convert the system time to the RTC time.

```
\> hwclock -w
```

Use the command ***hwclock –w YYYY-MM-DD hh:mm:ss*** to manually update the RTC time.

```
\> hwclock –w 2005-12-23 15:00:00
```

Use the command ***hwclock*** to query the updated RTC time.

```
\> hwclock
2005-12-23 15:00:00
```

Starting and Stopping Services

After booting up, the DA-660 computer runs several services continuously to serve requests from users or other programs. Notable services include telnet ("TEL0:"), console ("CON0:"), world wide web HTTP ("HTP0:"), and file transfer FTP ("FTP0:") etc. You seldom need to care about these services. However, you still can start up or stop a service with its associated name by using the command "services". For example,

Start the FTP service by

```
\> services start FTP0:
```

Stop the FTP service by

```
\> services stop FTP0:
```

Troubleshooting Network Connectivity

The **ipconfig** tool prints the TCP/IP-related configuration data of a host including the IP addresses, gateway and DNS servers.

```
\> ipconfig /all
Windows IP configuration
Ethernet adapter Local Area Connection:
IP Address: 192.168.4.127
Subnet Mask: 255.255.255.0
Adapter Name: IXP425ETHNPE2
Description: IXP425ETHNPE2
Adapter Index: 2
Address: 80 86 33 33 34 12
DHCP Enabled: NO
Ethernet adapter Local Area Connection:
IP Address: 192.168.14.202
Subnet Mask: 255.255.248.0
Default Gateway: 192.168.15.254
Adapter Name: IXP425ETHNPE1
Description: IXP425ETHNPE1
```

```
Adapter Index: 3
Address: 78 56 34 91 cc dd
DHCP Enabled: NO

Host name: DA660CE
Domain Name:
DNS Servers: 192.168.1.6
NODETYPE: 8
Routing Enabled: NO
Proxy Enabled: NO
```

To troubleshoot network connectivity, reachability, and name resolution, use the **ping** command. This command verifies IP-level connectivity to another TCP/IP computer by sending Internet Control Message Protocol (ICMP) Echo Request messages. The corresponding return Echo Reply messages are displayed, along with round-trip times. For more information, type **ping** without parameters.

```
\> ping www.moxa.com
Pinging Host www.moxa.com [192.168.1.16]
Reply from 192.168.1.16: Echo size=32 time<1ms TTL=126
Reply from 192.168.1.16: Echo size=32 time<1ms TTL=126
Reply from 192.168.1.16: Echo size=32 time<1ms TTL=126
```

The **route** utility allows you to view or modify network routing tables. Type this command without parameters to view a list of functions.

```
\> route
```

To view current routing items in the tables,

```
\> route PRINT
```

To add a routing item on network interface 1,

```
\> route ADD 192.168.0.0 MASK 255.255.0.0 192.168.15.254 IF 2
```

To delete a routing item,

```
\> route DELETE 192.168.0.0
```

Simple Network Management Protocol (SNMP)

SNMP is the Internet Standard protocol for network management and part of the TCP/IP protocol suite. SNMP was developed to monitor and manage networks. It uses a distributed architecture that consists of agents and managers:

SNMP Agent

The SNMP agent is an SNMP application that monitors network traffic and responds to queries from SNMP manager applications. The agent also notifies the manager, by sending a trap, when significant events occur.

SNMP Manager

An SNMP manager is an SNMP application that generates queries to SNMP-agent applications and receives traps from SNMP-agent applications.

The DA-660 computer installs an SNMP agent to serve as an SNMP device. You should install the SNMP manager on the workstation computer (for example, a Linux system) that monitors the network. After installing the nodes, you need to configure the SNMP manager and agent.

To check SNMP agent capabilities in a target computer (e.g, network IP at 192.168.3.127), please logon the workstation computer that, for example, a Linux-based computer, the SNMP manager resides and type

```
\> snmpwalk -v 2c -c public 192.168.3.127 system
SNMPv2-MIB::sysDescr.0 Microsoft Windows CE Version 5.0 (Build 1400)
SNMPv2-MIB::sysObjectID.0 SNMPv2-SMI::enterprises.8691.13.7420
SNMPv2-MIB::sysUpTime.0 1282929
SNMPv2-MIB::sysContact.0 Your System Contact Here
SNMPv2-MIB::sysName.0 WindowsCE
```

You will see a series of messages from the SNMP agent. From there, you can monitor and manage the computer.

3

Web-based Management System

Note: Internet Explorer 5.5 or above is required to use the web based management system.

DA-660 ready-to-run embedded computers are network-centric platforms designed to be used as front-end computers for data acquisition and industrial control. Due to the distributed characteristics of the devices that these computers control, they often reside in harsh areas as the devices themselves and are away from system administrators. To manage these computers, operations such as networking/server configuration, file management, and process (thread) monitoring/control become a critical area to consider.

A web-based management system is installed on the DA-660 computer to resolve these management issues, and to reduce the amount of work required by system administrators. The system incorporates often-used features in the CGI and categorizes using a menu bar.

Before operating the system, make sure you have a network connection from your PC to the target computer, and verify that you can open an Internet browser from your PC after the connection is established. Then, use the IP address of the target computer as a home page URL. After the main screen is displayed, click **Web-Based Management**.

Input your authentication data, including user ID and password, in the corresponding fields (both the ID and password are case sensitive) and then press Enter to request access to the management system.

The default User ID and Password are:

User ID: admin

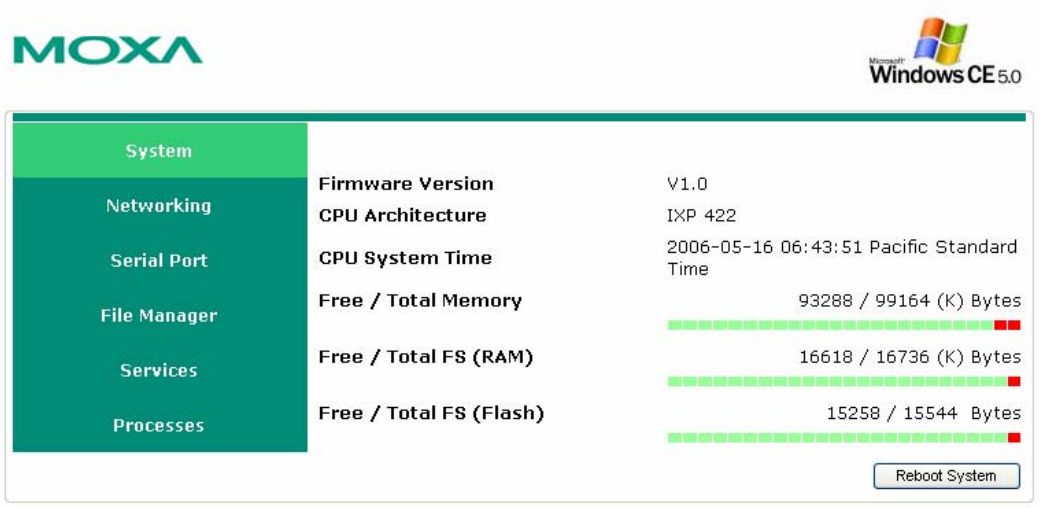
Password: admin

The following topics are covered in this chapter:

- **System Information**
- **Netowrking/Server Configuration**
- **Serial Port Configuration**
- **Process (Thread) Monitoring/Control**
- **Launching Processes Automatically**
- **Services Monitoring/Control**
- **Binary/Text File Management**

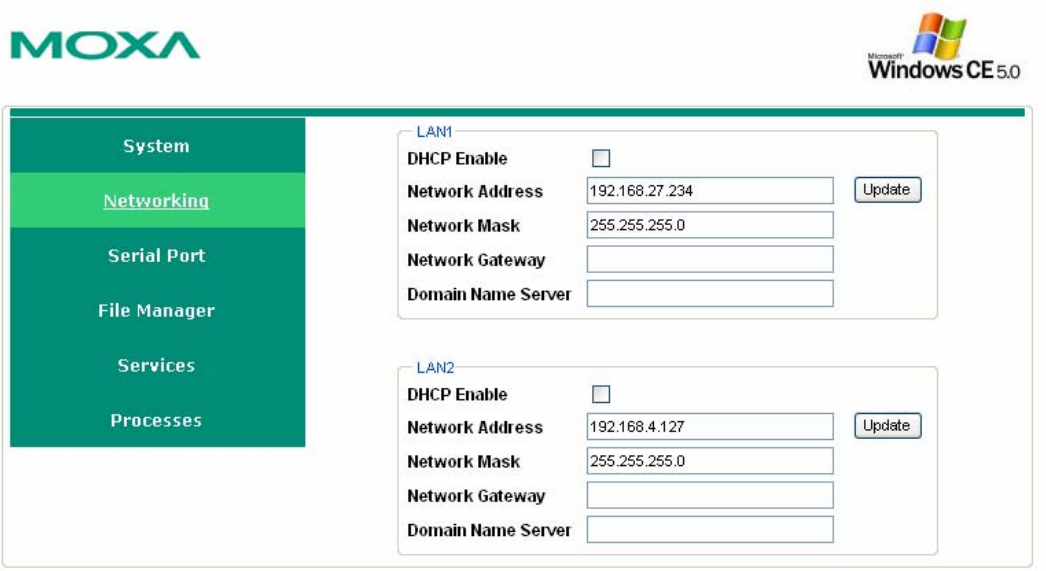
System Information

After you logon successfully, the main page displays the system information of the target computer, including the firmware version of the computer, the CPU system time, and system resources including main memory and file system usage (RAM and Flash).



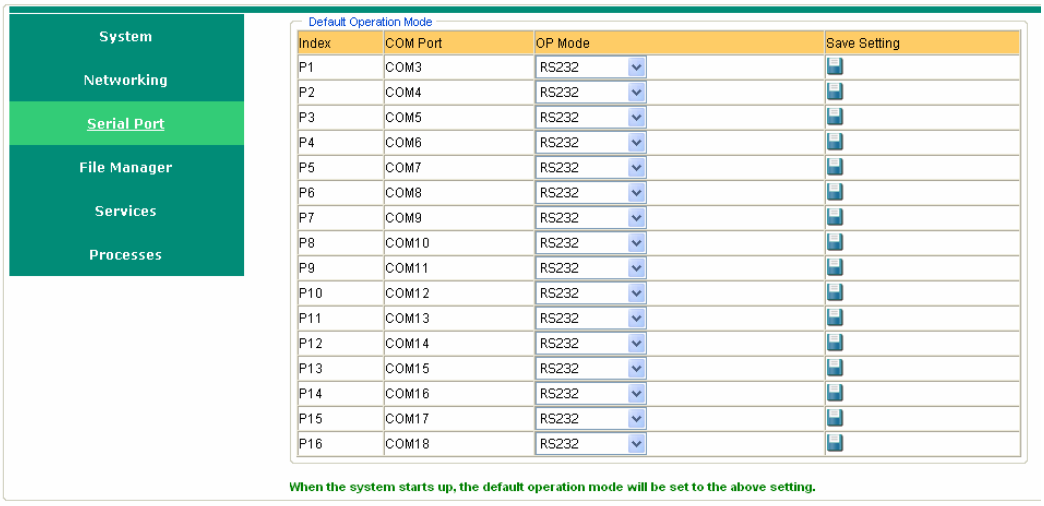
Networking/Server Configuration

The target computer has two network interfaces. To view or change their settings, click the ***Networking*** item on the menu bar. After the page loads, enter the relevant details on the corresponding text fields and then click "**Update**" to make the changes effective for the interfaces.



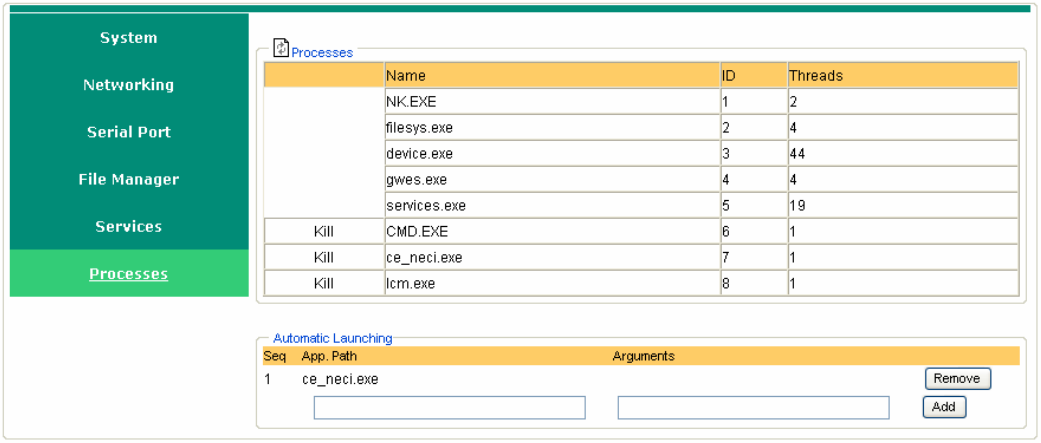
Serial Port Configuration

The target computer consists of multiple high-performance serial ports, each with 3-in-1 (RS232, RS422 or RS485) operation mode switch capability, with the default at RS232. You can specify each of these ports to operate at a specific mode. The updated settings take effect after the system is rebooted and remain in force until another update is made.



Process (Thread) Monitoring/Control

At runtime, the target computer manages up to 32 applications that you can monitor and control via the management system. To view current processes, click the ***Processes*** item on the main menu bar. You can kill a process by clicking the **kill** button next to the process name.



Launching Processes Automatically

To have your application start on boot, do the following:

Step 1: Click the ***Processes*** item on the main menu bar. At the lower part of the page, there is an area marked as "Automatic Launching".

Step 2: Fill in the full path of the application in the first text field and its arguments in a separate text field if there are any.

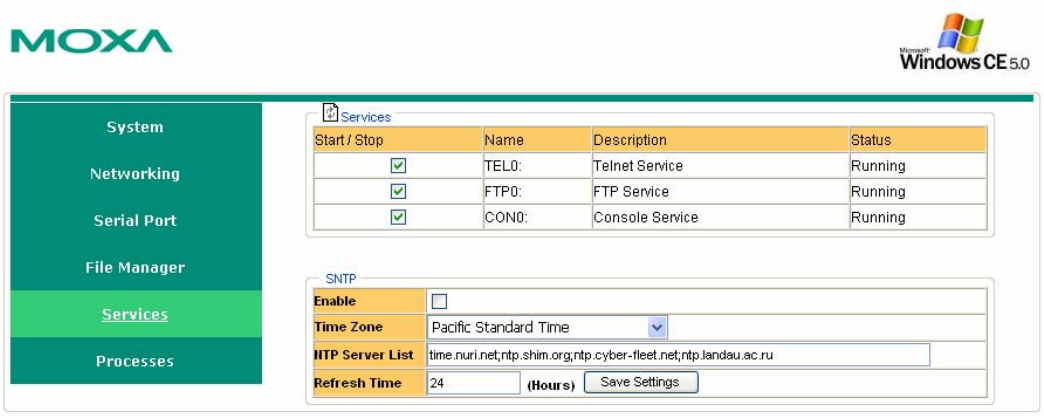
Step 3: Click **Add**.

Services Monitoring/Control

Some services run in the background to provide services, such as ftp, and telnet daemons etc, for user requests. To monitor and control these services, do the following:

Step 1: Click the **Services** item on the main menu bar. The running services are displayed.

Step 2: Click the relevant check box to toggle a start/stop operation for the desired service.

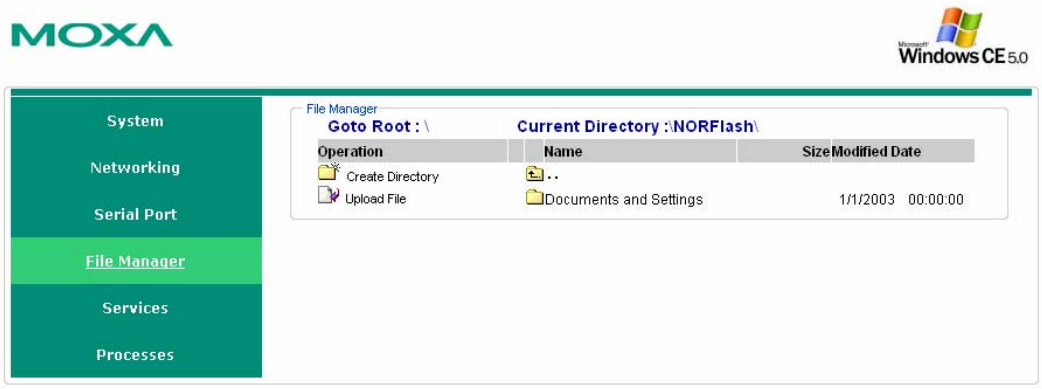


Some listed services cannot be stopped in order to maintain normal operation of the computer. Such services do not have a check box next to them.

Binary/Text File Management

On a PC, it is certainly convenient to have a friendly window-based file manager to browse, delete, and organize files and directories. On the target computer, such a convenient feature is simulated by the web-based management system. Click **File Manager** to view the directory tree of your target computer. Using the file manager, you can perform the following operations:

- To browse a child directory, click the name of the directory.
- To delete a file, click the ***X*** in front of the file icon.
- To create a child directory, click ***Create Directory*** and then follow the instructions.
- To refresh the current directory, click ***Current Directory*** at the top of the page.



In addition, the management system offers a mechanism for file upload. This mechanism helps you transfer files from your workstation to the target computer in an easy way. For instance, after you have built an application on the development workstation, you can use this mechanism to upload the application to the current directory of the target computer.

Step 1: Click **Upload File**. A browser window pops up.

Step 2: From the pop-up browser window, click **Browse** to bring up a local file manager.

Step 3: Browse to and select the file that you want to upload and click **Open**.

Step 4: Navigate back to the browser window, and click **OK**. The system starts to upload the file.

Step 5: After the file is uploaded completely, refresh the page.

4

Application Development

Application development on the DA-660 computer can be accomplished with a number of well-known tools that are provided by the Windows® environments on programmers' workstations. These tools are easy for Windows® programmers to use. This chapter presents the development tools for Windows® Embedded Application Development.

The following topics are covered in this chapter:

- **Installing Development Tools (IDEs)**
  - VB.NET/C# Applications: Using Visual Studio 2005
  - C/C++ Applications: Using eMbedded Visual C++ (eVC) 4.0
- **Installing MOXA Windows® CE 5.0 C/C++ SDKs**
- **Developing Applications with IDEs**
  - Developing eMbedded Visual C++ 4.0 applications with an imported SDK
  - Developing VB.NET / C# applications with a .NET Compact Framework
- **Visual Studio 2005 Remote Tools**
  - Using Visual Studio C#, VB.Net or C++ 2005 Express
  - Remote Tool # 1: Application Debugger
  - Remote Tool #2: Remote File Explorer

Installing Development Tools (IDEs)

Building applications can be done with a number of friendly Integrated Development Environment (IDE) tools. Choose the tools based on the application language you plan to use and install them on your development workstation.

VB.NET/C# Applications: Using Visual Studio 2005

Microsoft® Visual Studio 2005 is a complete set of development tools for building ASP.NET Web applications, XML Web services, desktop applications, and mobile applications. Visual Basic, Visual C++, Visual C#, and Visual J# all use the same IDE, which allows them to share tools and facilitates the creation of mixed-language solutions.

Microsoft also offers Visual Studio 2005 Express for student developers. It is free and can be downloaded from MSDN's web site.

Note: When you have installed the tools on your workstation computer, you can start developing your applications. There is no need to install any other SDK.

C/C++ Applications: Using eMbedded Visual C++ (eVC) 4.0

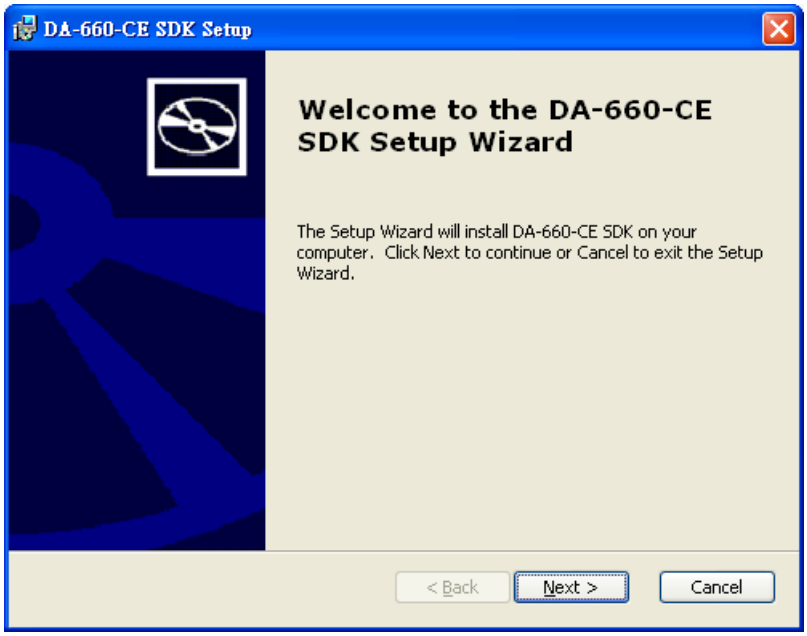
The eVC 4.0 tools are free. You can download them from MSDN's eMbedded Visual Tools Download Page. Install the eVC 4.0 tools, and then install the service pack in the following order:

- Install eMbedded Visual C++ 4.0 (230 MB)
- Install Service Pack 4 for eVC 4.0 (68 MB)
- Install MOXA Windows® CE 5.0 C/C++ SDKs

Installing MOXA Windows® CE 5.0 C/C++ SDKs

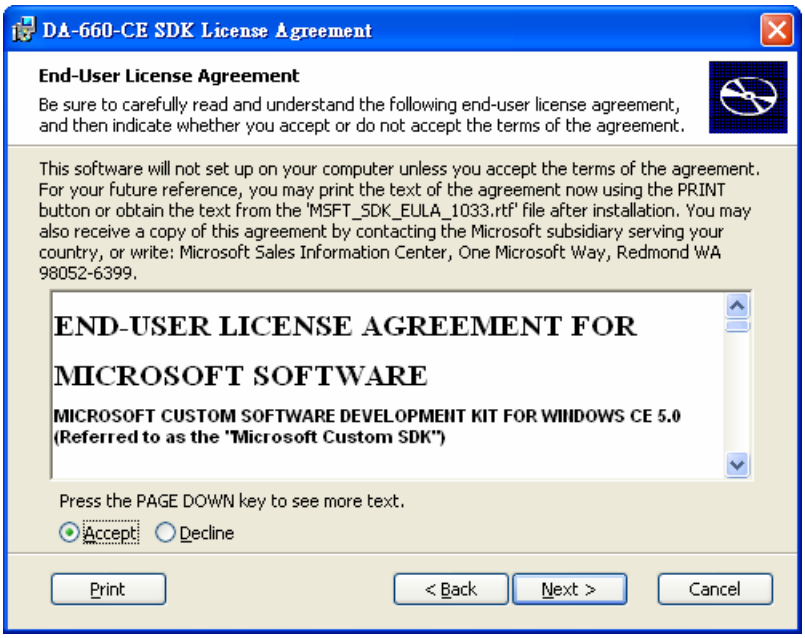
After installing eVC 4.0 on the development workstation, you need to install an SDK from MOXA. The SDK file, e.g., ***DA660CESDK1_0.msi,*** is available on the package CD.

1. Double click on the MSI file. A Setup Wizard pops up to start the installation process. Choose **Next.**

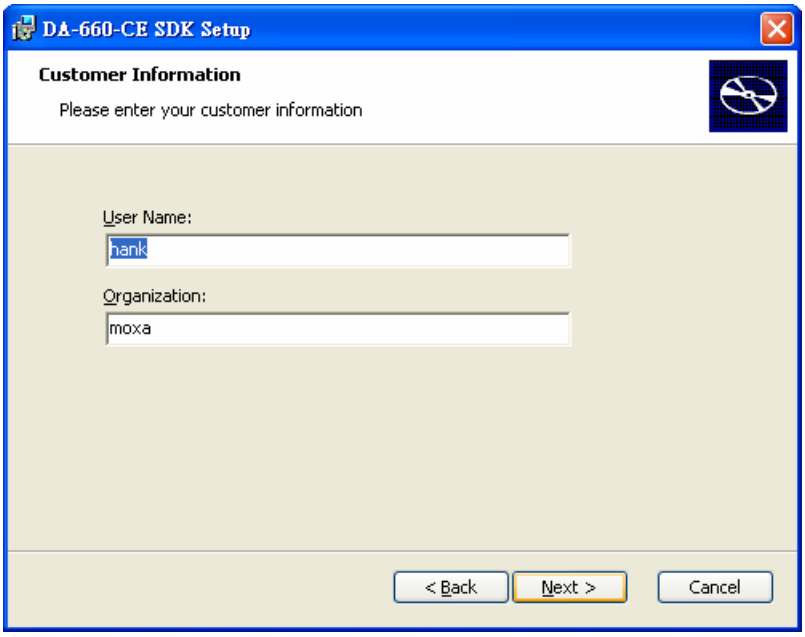

DA-660-CE SDK Setup
Welcome to the DA-660-CE SDK Setup Wizard
The Setup Wizard will install DA-660-CE SDK on your computer. Click Next to continue or Cancel to exit the Setup Wizard.
< Back
Next >
Cancel

   If an error message appears, click **Close** to continue the installation process.

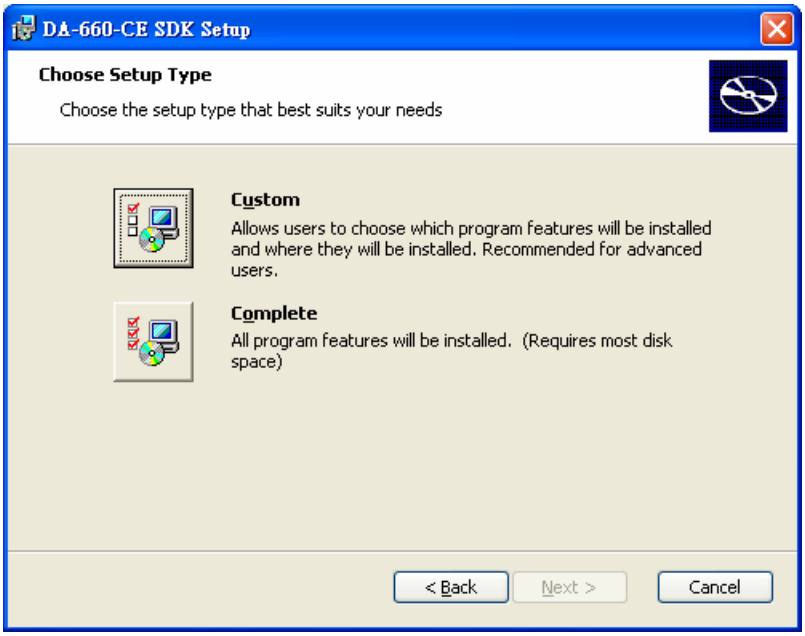
2. Read the "License Agreement" dialog, and if you accept the terms of the end-user license agreement (EULA), choose **Accept**, and then click **Next**.


DA-660-CE SDK License Agreement
End-User License Agreement
Be sure to carefully read and understand the following end-user license agreement, and then indicate whether you accept or do not accept the terms of the agreement.
This software will not set up on your computer unless you accept the terms of the agreement. For your future reference, you may print the text of the agreement now using the PRINT button or obtain the text from the 'MSFT_SDK_EULA_1033.rtf' file after installation. You may also receive a copy of this agreement by contacting the Microsoft subsidiary serving your country, or write: Microsoft Sales Information Center, One Microsoft Way, Redmond WA 98052-6399.
END-USER LICENSE AGREEMENT FOR MICROSOFT SOFTWARE
MICROSOFT CUSTOM SOFTWARE DEVELOPMENT KIT FOR WINDOWS CE 5.0
(Referred to as the "Microsoft Custom SDK")
Press the PAGE DOWN key to see more text.
Accept
Decline
Print
< Back
Next >
Cancel

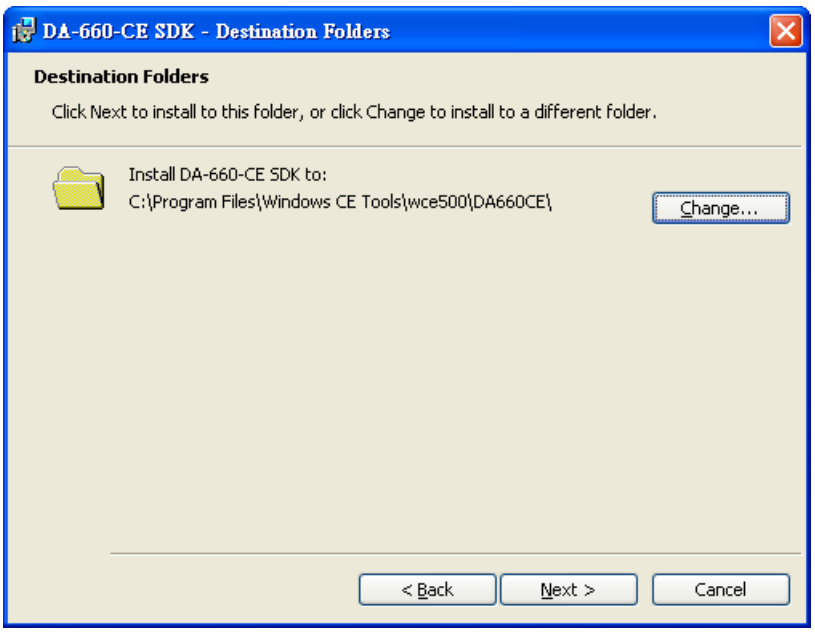
3. Type your name for **User Name**, the name of your company for **Organization**, and then click **Next**.



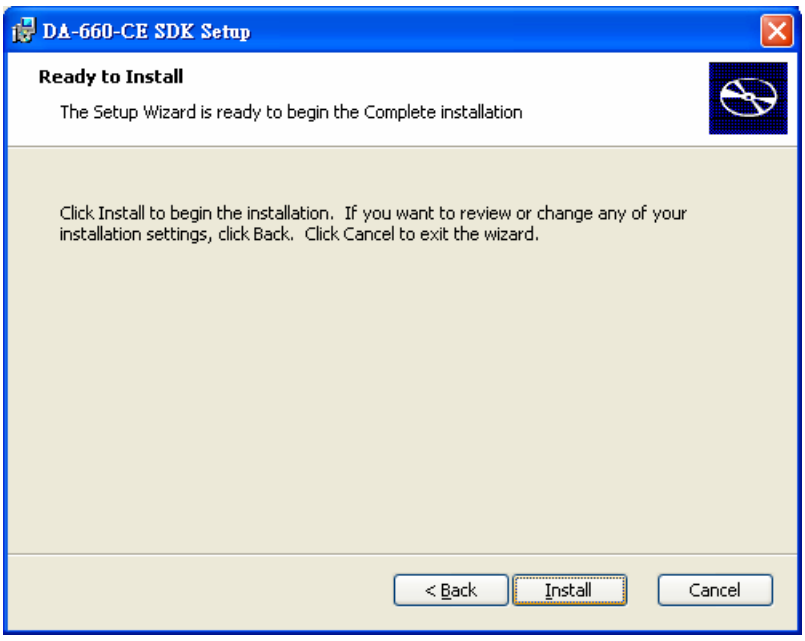
4. To install all functionality provided with the SDK, click **Complete** and then click **Next**.


DA-660-CE SDK Setup
Choose Setup Type
Choose the setup type that best suits your needs
Custom
Allows users to choose which program features will be installed and where they will be installed. Recommended for advanced users.
Complete
All program features will be installed. (Requires most disk space)
< Back
Next >
Cancel

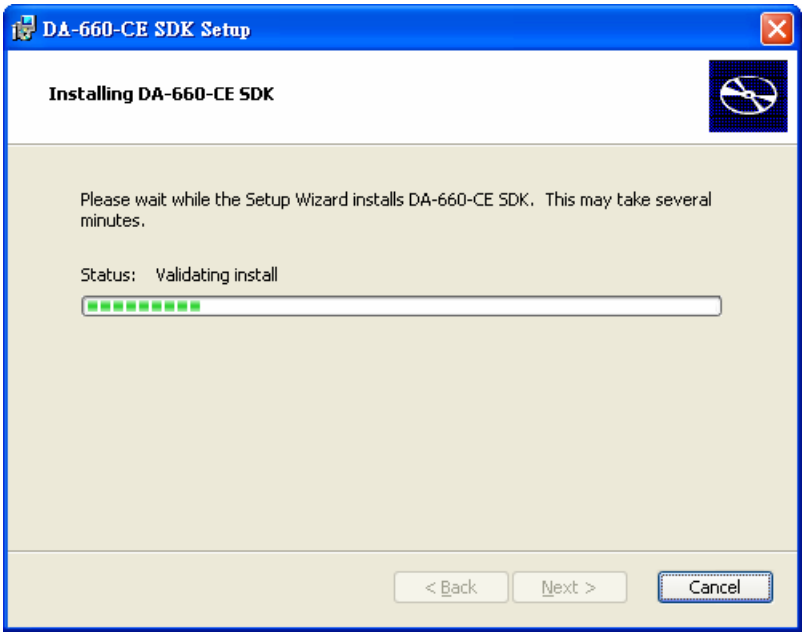
5. If you have a folder preference where you want the SDK to reside, click **Change** and browse to the folder. Otherwise, place it in the default folder and then click **Next**.


DA-660-CE SDK - Destination Folders
Destination Folders
Click Next to install to this folder, or click Change to install to a different folder.
Install DA-660-CE SDK to:
C:\Program Files\Windows CE Tools\wce500\DA660CE\
Change...
< Back
Next >
Cancel

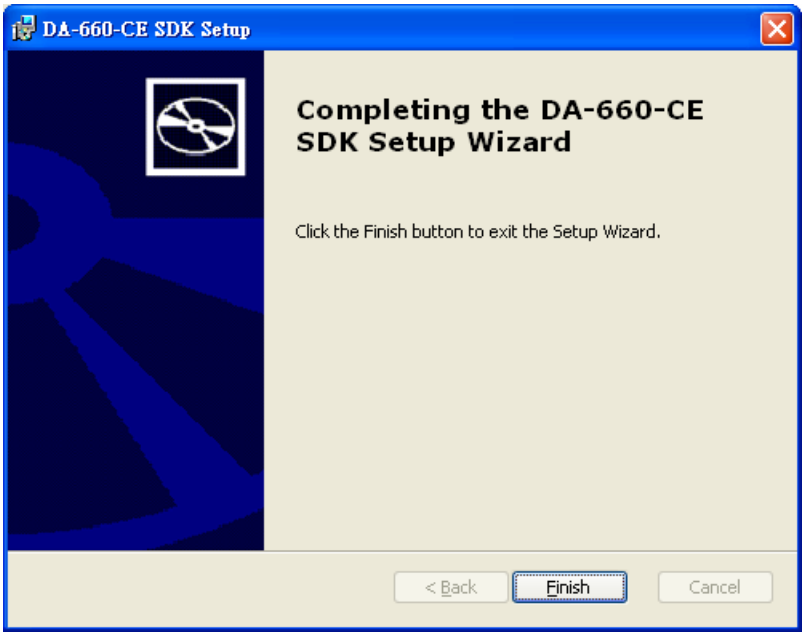
6. Click **Install** to install the SDK.


DA-660-CE SDK Setup
Ready to Install
The Setup Wizard is ready to begin the Complete installation
Click Install to begin the installation. If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.
< Back
Install
Cancel

7. Wait for the installation to be completed.


DA-660-CE SDK Setup
Installing DA-660-CE SDK
Please wait while the Setup Wizard installs DA-660-CE SDK. This may take several minutes.
Status: Validating install
< Back
Next >
Cancel

8. Once the installation is complete, click **Finish** to close the installer.

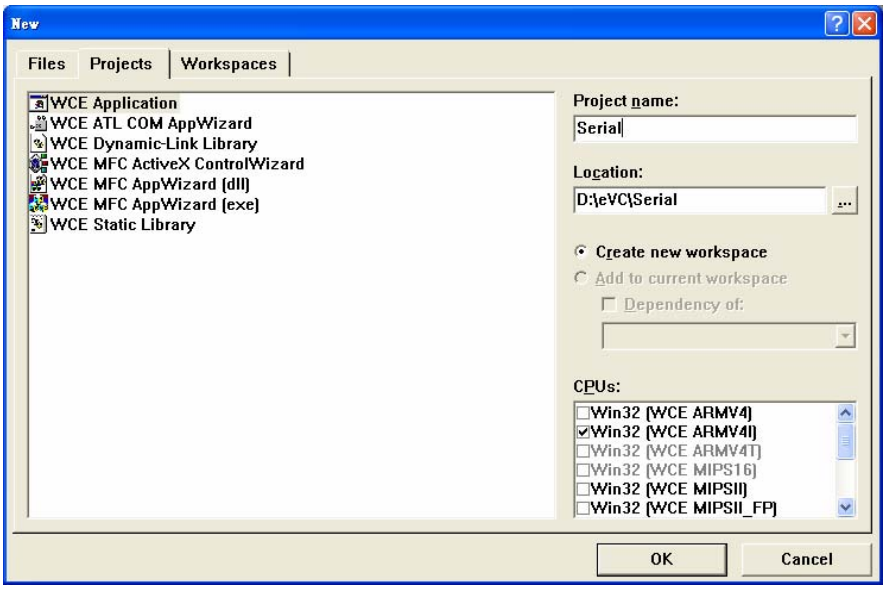


Developing Applications with IDEs

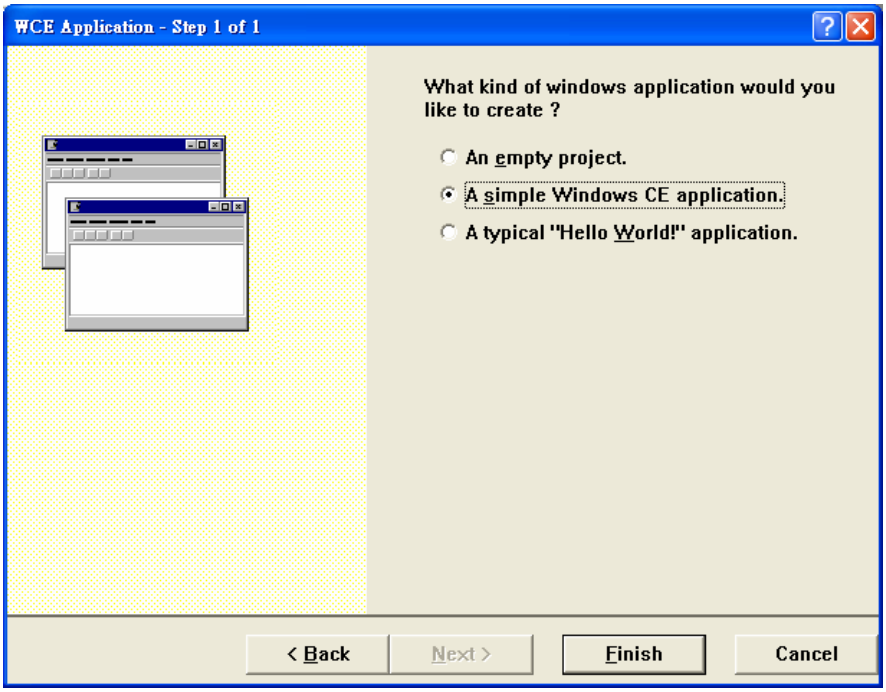
Windows® CE 5.0 with the MOXA SDK you have just installed offers application developers the familiar Windows® IDE and application programming interface (API) with the versatility of scripting languages. We use eVC 4.0 and Visual Studio .Net 2005 as examples to demonstrate how easy it is to develop applications.

Developing eMbedded Visual C++ 4.0 applications with an imported SDK

1. Start Microsoft® eMbedded Visual C++ 4.0.
2. From the **File** menu, choose **New**.
3. Choose the **Projects** tab and then select the type of application that you want to build.

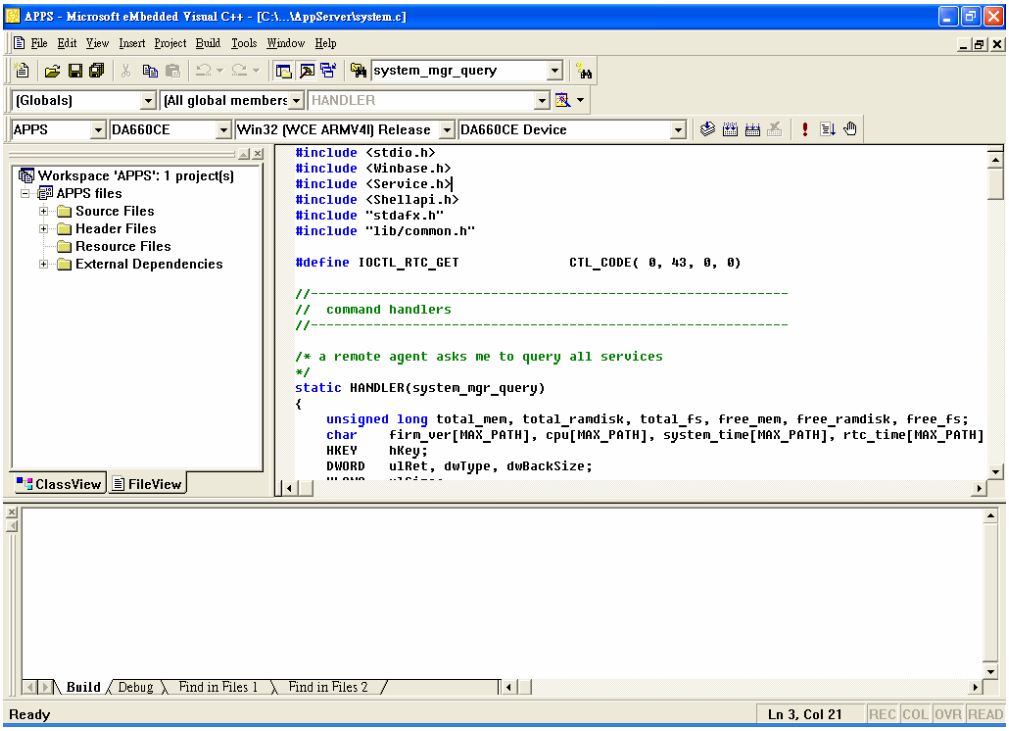


4. Fill in the project name and click **OK**.
5. Choose the type of application that you want to create and then click **Finish**.


WCE Application - Step 1 of 1
What kind of windows application would you like to create ?
An empty project.
A simple Windows CE application.
A typical "Hello World!" application.
< Back
Next >
Finish
Cancel

6. From the **Build** toolbar, choose the SDK (***DA660CE*** in this example), the type of run-time image (Release or Debug), and the target device (***DA660CE*** Device in the example).

7. Develop your application.



8. From the **Build** menu, choose **Rebuild All** to compile your application.
9. Ignore the following dialog if it pops up. This is generated by ActiveSync while developing a mobile application. The DA-660 computer is not a mobile device.

10. After you finish developing your application, use the web-based management system to upload it to the target computer.
11. Log on to the target computer and execute the program at the console prompt.

Developing VB.NET / C# applications with a .NET Compact Framework

1. Open Microsoft® Visual Studio .Net 2005.
2. From the **File** menu, choose **New → Project.**
3. Choose the **Project Type** and then select the **Smart Device Application** as the type of project.
4. Fill in the project name and click **OK**.
5. Choose **Windows CE** as the target platform.
6. Select the desired project type and click **OK**.

7. Develop your application.
8. From the **Device** toolbar, choose **Windows CE.Net Device**.
9. From the **Build** menu, choose **Build Project** or **Rebuild Project**.
10. When you complete your application, use the web-based management system to upload it to the target computer.
11. Log on to the target computer. At the console prompt, execute your application directly, if it is written in C#. Otherwise, if it is a VB script or java script application, execute it using the program **wcescript.exe**.

Visual Studio 2005 Remote Tools

Microsoft provides many powerful utilities, such as Remote Debugger, Remote File Explorer, Remote Registry Editor, Remote Process Viewer and Remote Spy, for Windows CE application development and management.

Before using these tools, you need to install 6 executable files, namely Clientshutdown.exe, CMAccept.exe, ConmanClient2.exe, DeviceDMA.dll, eDbgTL.dll and TcpConnectionA.dll, to the target computer, DA-660. Find these 6 files under the ***Windows*** directory on the target computer. If the files are missing, find them from the path "\utility\vs2005" of the package CD and upload them to the target computer.

Before using any of these tools, log on to the target computer and execute the programs **ConmanClient2** and **CMAccept**:

>\ **ConmanClient2**

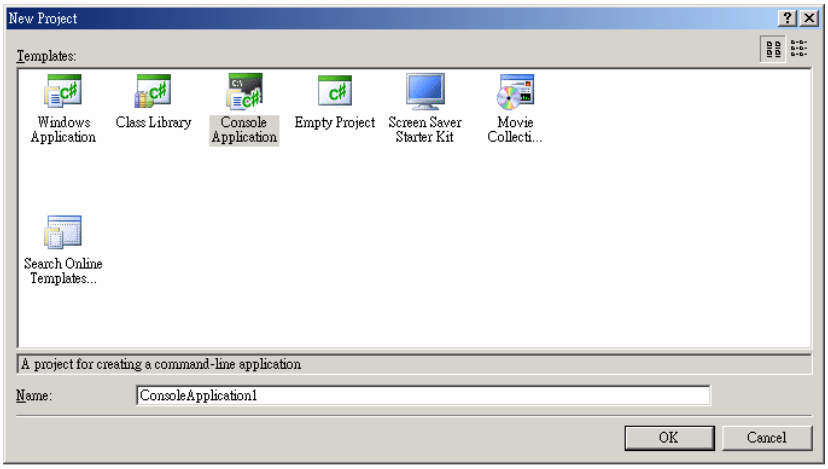
>\ **CMAccept**

The program **CMAccept** exits after a remote connection is made. If you want to make a new connection, run the program again.

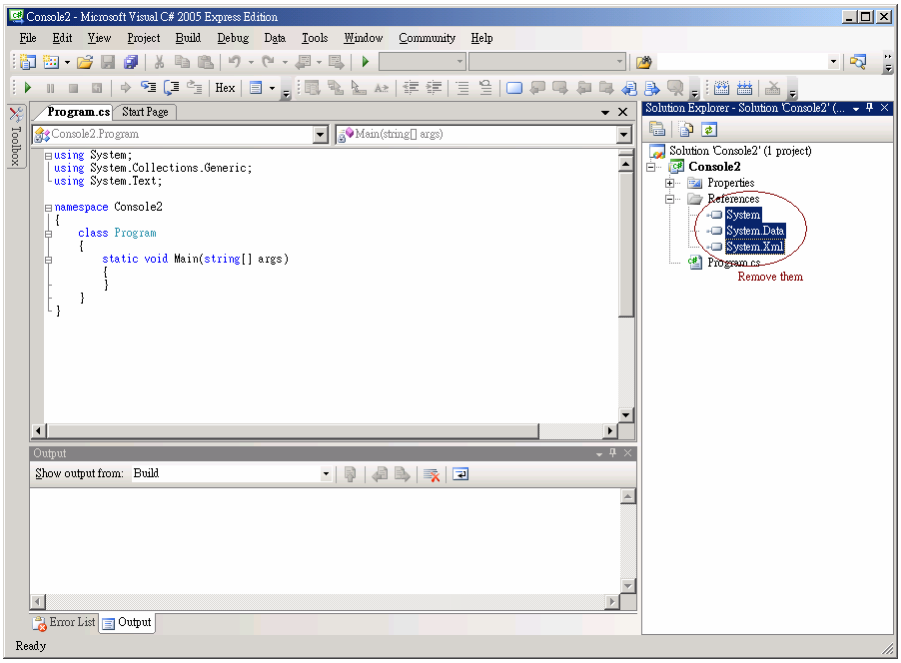
Using Visual Studio C#, VB.Net or C++ 2005 Express

1. Download and install Visual Studio 2005 Express from the Microsoft web site, http://msdn.microsoft.com/vstudio/express/.
2. Make sure you have already installed the .Net Framework 2.0 SDK and .Net Compact Framework 2.0. Check here: http://www.microsoft.com/downloads/details.aspx?FamilyID=FE6F2099-B7B4-4F47-A244-C96D69C35DEC&displaylang=en.
3. Start Microsoft® Visual Studio 2005 Express.

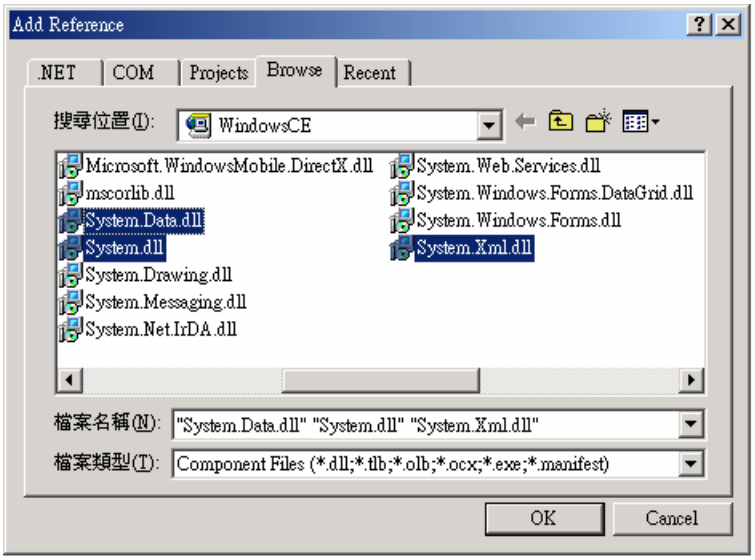
4. From the **File** menu, choose **New → Project.**



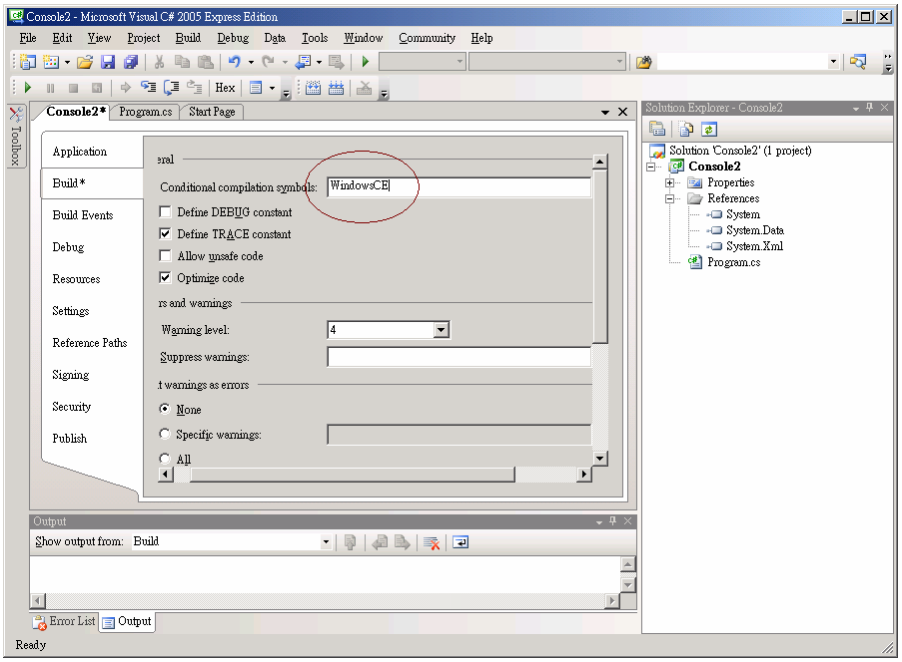
5. Remove the original reference library



6. Add Windows CE 5.0 .Net Class DLL to the References from **C:\Program Files\Microsoft.NET\SDK\CompactFramework\v2.0\WindowsCE**.



7. Set up the Windows CE environment variable from **Project → Project Properties**.



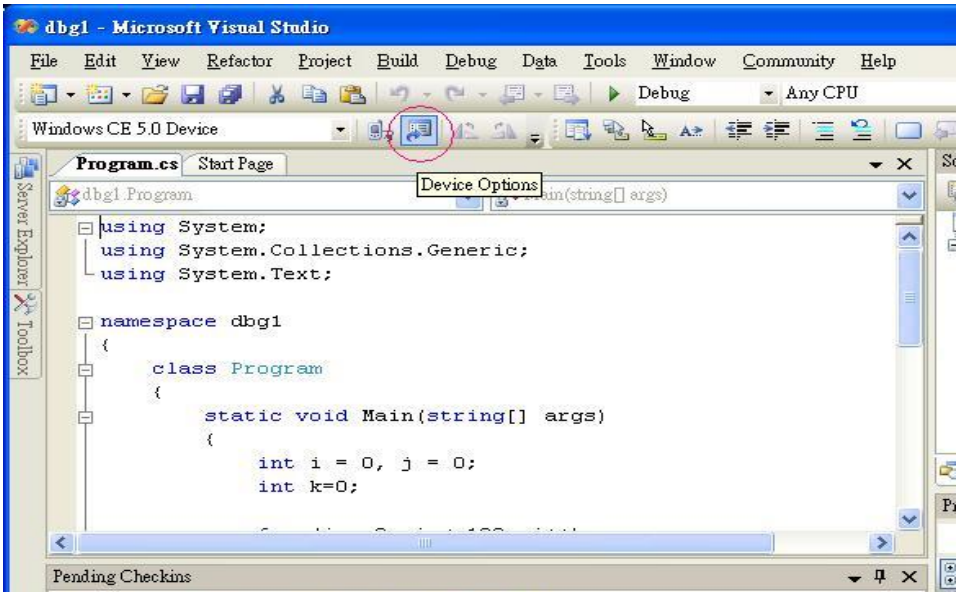
8. From now on, you can start developing your programs. Note that Visual Studio 2005 Express does not support on-line debugging.

Remote Tool # 1: Application Debugger

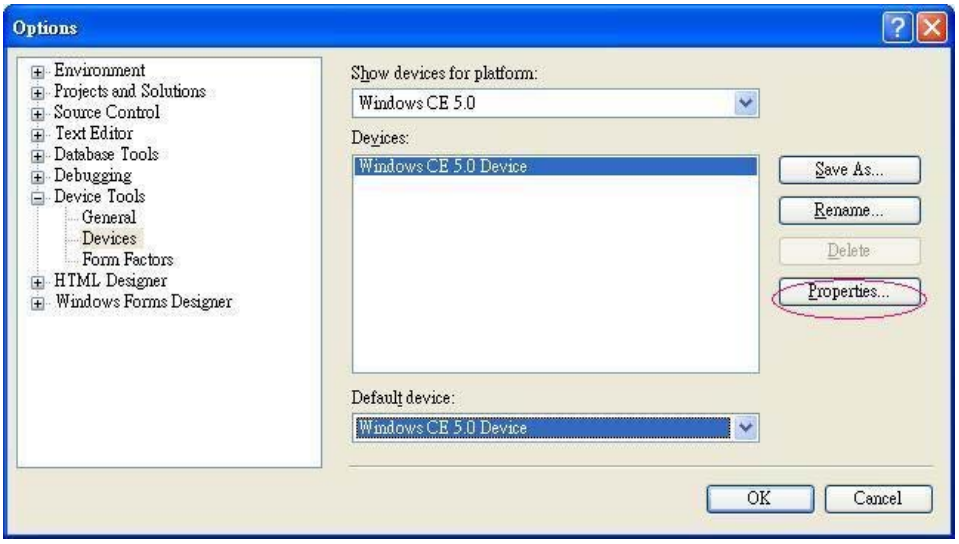
Visual Studio 2005 provides a friendly utility to help developers debug applications faster and easier. This is a step-by-step debugging method, just like the ones available with most PC-based development tools. Take the following steps to create a debugging session.

1. Open a Visual Studio 2005 Smart Device project.

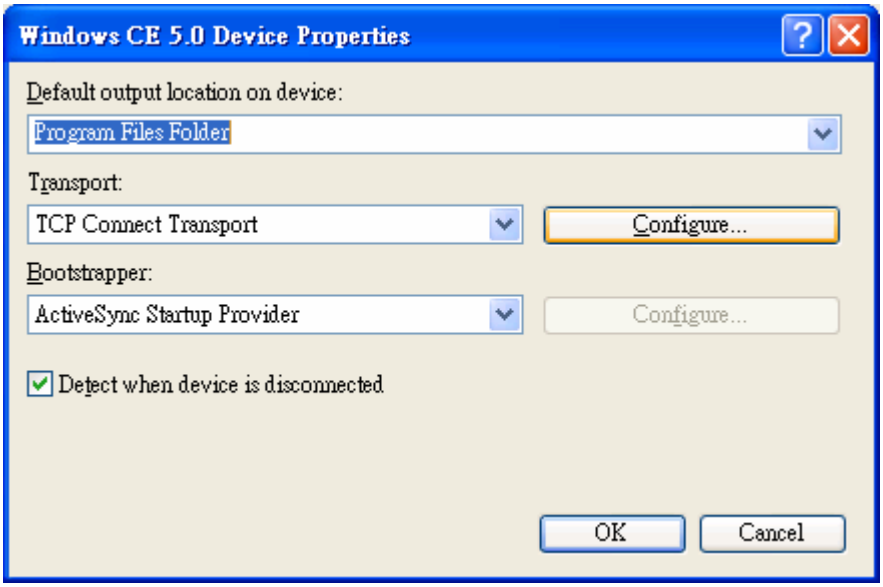
2. Click **Device Option**.



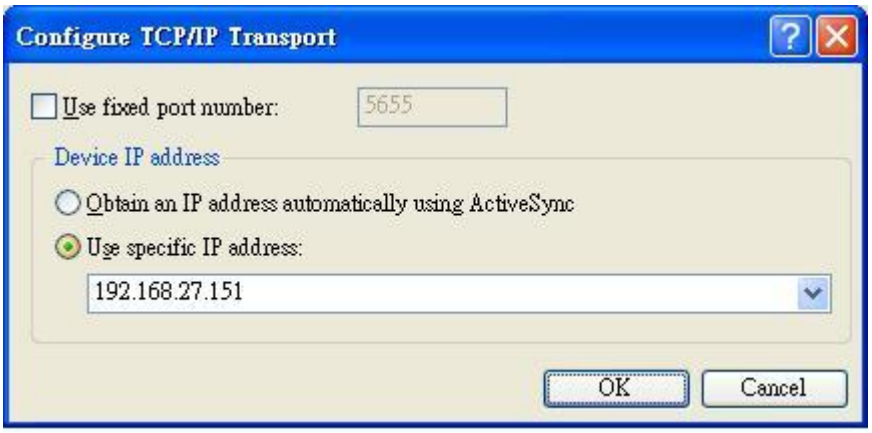
3. Click **Properties**… from the Options pop-up window.



4. Click **Configure…** from the Windows CE 5.0 Device Properties pop-up window.

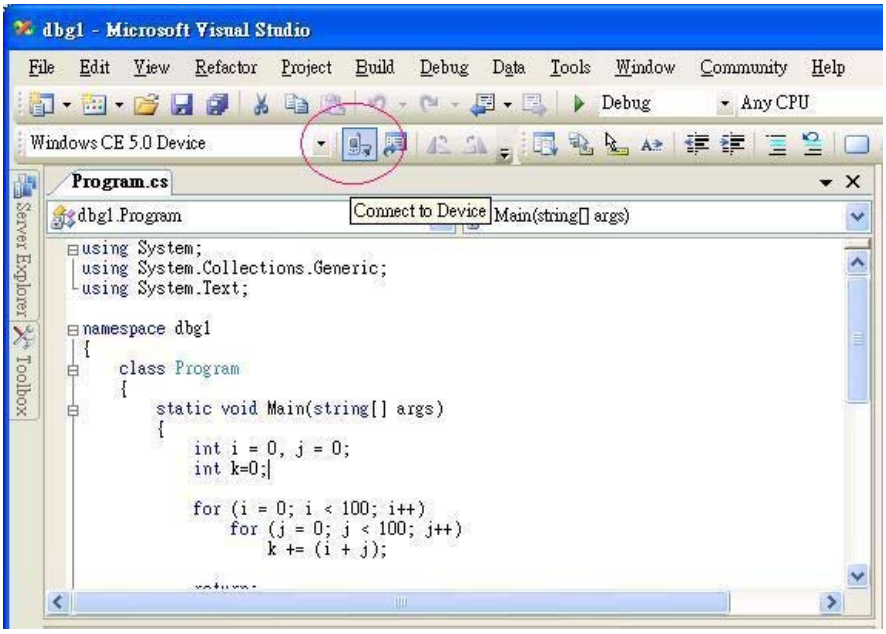


5. Select the **Use specific IP Address** option and enter the IP address of the target computer.



6. Click **OK** to go back to the main screen.

7. Click **Connect to Device**.



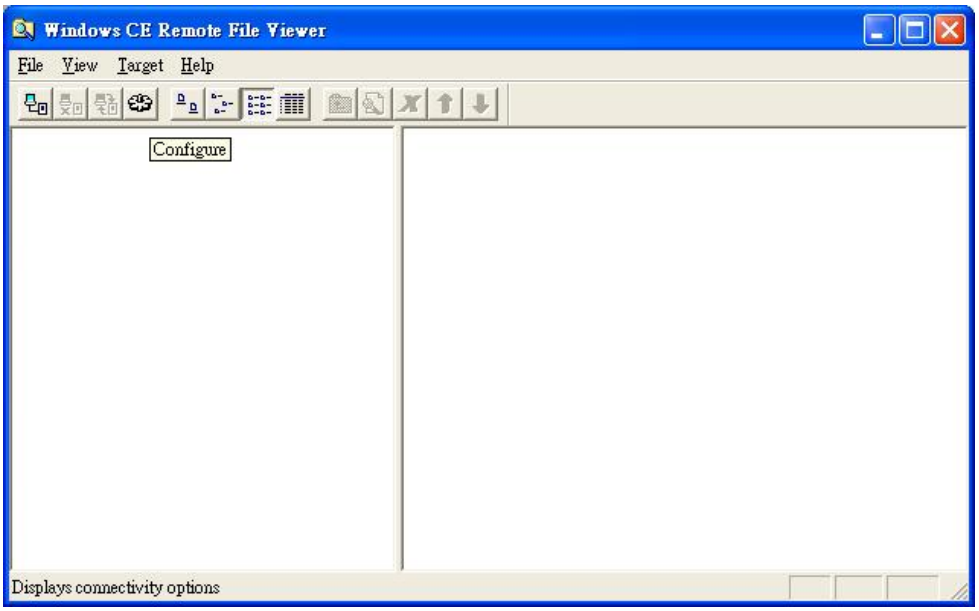
8. Wait for the connection to be made.

9. Start the debugging process.

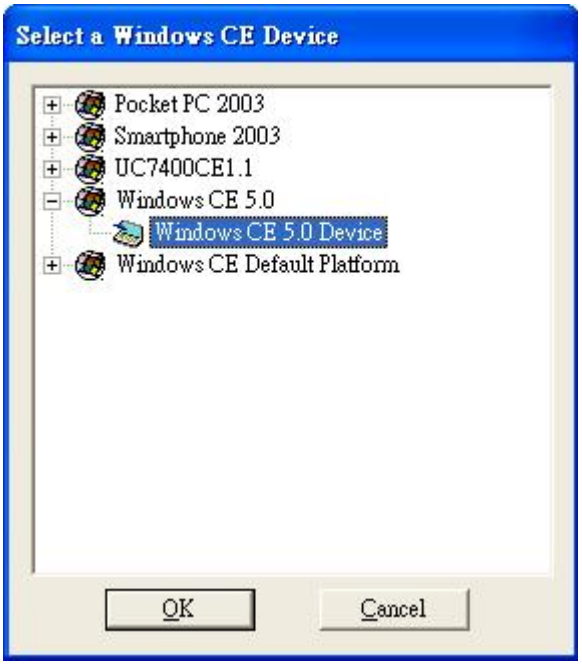
Remote Tool #2: Remote File Explorer

Assume that you have configured a remote connection to the target computer by following the steps described above.

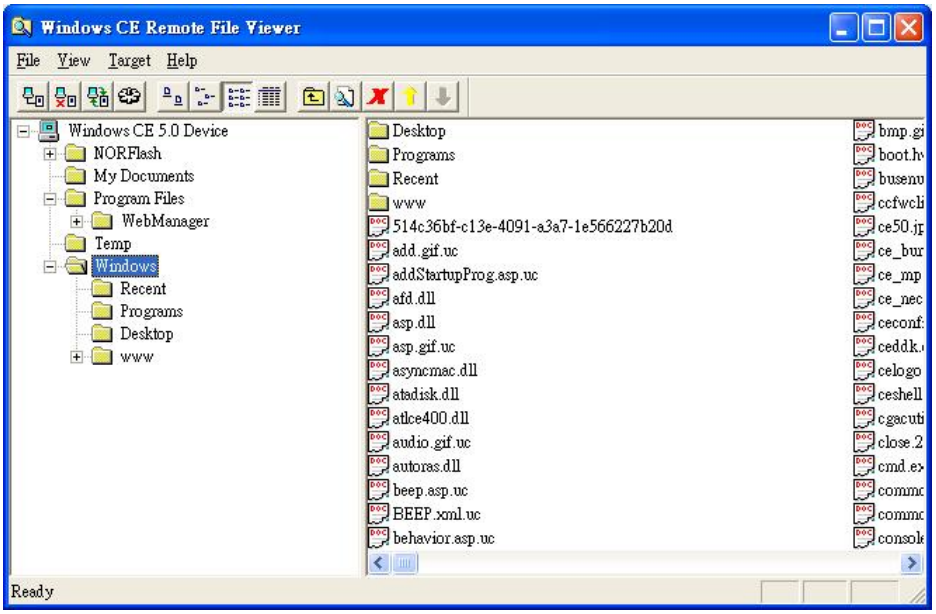
1. Launch the Remote File Viewer.



2. Click the Windows CE 5.0 Device and then click **OK**.



3. Start using the tool to navigate files and directories.



5

Programming Examples

Embedded Visual C++4 Examples

The following examples are designed to ease application development. Find and copy useful code segments to your C/C++ programs. In addition, to simplify application development with the eVC4.0 tool, we provide you with a device library (***da660.lib***). To link the library with your eVC4.0 compilation environment, perform the following steps from your eVC4.0 environment:

- From the main tool bar, choose **Project → Settings**.
- From **Project Settings**, select the **Link** tab.
- Append ***da660.lib*** to the text field Object/Library Modules.
- Click **OK**

Before you compile your application, please make sure that the header file and the library file in the SDK directories are as follows. If any of them is not in the specified directory, find it on the package CD and copy it to the specified directory.

"C:\Program Files\Windows CE Tools\wce500\DA660CE\Include\Armv4i\moxa\devices.h"
"C:\Program Files\Windows CE Tools\wce500\DA660CE\Lib\ARMV4I\da660.lib"

Example #1—MOXA UART Supporting RS-232/422/485

The following C/C++ code shows a sample application transmitting data from port "COM3" to port "COM4" using the RS-232 operation mode. After these ports have been opened, the application generates a thread to receive data from port "COM3" and then the application itself executes as a main thread to transmit data to port "COM4".

```
#include "stdafx.h"
#include <stdio.h>
#include <moxa/devices.h>

#define MAX_DATA_LEN 128

//=====================================================================
static HANDLE createComHandle(WCHAR  *comPort, unsigned int baudrate)
{
  HANDLE hCom;
  DCB dcb;
  COMMTIMEOUTS         to;

  hCom = CreateFile(comPort,  GENERIC_READ | GENERIC_WRITE,
           0, // exclusive access, until the handle is closed
           NULL, // no security
           OPEN_EXISTING,
           0, // no overlapped I/O
           NULL); // null template
  if (hCom==INVALID_HANDLE_VALUE )
           return NULL;

  // set serial setting
  GetCommState(hCom, &dcb);
  dcb.BaudRate = baudrate;
  dcb.ByteSize = 8;
  dcb.StopBits = ONESTOPBIT;
  dcb.Parity = NOPARITY;
  dcb.fRtsControl = RTS_CONTROL_HANDSHAKE;
  dcb.fOutxCtsFlow = TRUE;
  SetCommState(hCom, &dcb);

  // set timeout parameter
  to.ReadIntervalTimeout = 0;
  to.ReadTotalTimeoutMultiplier = 0;
  to.ReadTotalTimeoutConstant = 0;
  to.WriteTotalTimeoutMultiplier = 0;
  to.WriteTotalTimeoutConstant = 0;
  if (!SetCommTimeouts(hCom,&to))
  {
           printf("SetCommTimeouts error!\n");
```

```
            CloseHandle(hCom);
            return NULL;
  }
  return hCom;
}

//====================================================================
static DWORD comReadThread(LPVOID param)
{
  HANDLE hCom = (HANDLE)param ;
  DWORD rtn;
  unsigned char buffer[MAX_DATA_LEN+1];
  while(1)
  {
            if (ReadFile(hCom, buffer, MAX_DATA_LEN, &rtn, NULL)==0)
            {
                    printf("read data fail\n");
                    return 0;
            }
            buffer[rtn] = '\0';
            printf("Data = %s\n", buffer);
  }
  CloseHandle(hCom);
}

int
comPair(WCHAR *wComPort, WCHAR *rComPort, unsigned int baudrate)
{
  HANDLE wCom, rCom;
  WCHAR sPort[64];
  DWORD rtn, i, loop=0;
  unsigned char buffer[MAX_DATA_LEN];
  HANDLE waitH = CreateEvent(NULL, FALSE, FALSE, NULL);

  if (waitH==NULL)
            return 99;

  /* create a handle to port "COM3" for transmitting data */
          wsprintf(sPort, L"$device\\%s\0", wComPort);
  wCom = createComHandle(sPort, baudrate);
  if (wCom==NULL)
  {
            printf("Fail to create write port\n" );
            return 1;
  }
  /* create a handle to port "COM3" for receiving data */
  wsprintf(sPort, L"$device\\%s\0", rComPort);
  rCom = createComHandle(sPort, baudrate);
```

```
 if (rCom == NULL)
 {
          printf("Fail to create read port\n");
          CloseHandle(wCom);
          return 2;
 }
 /* for a thread to handle receiving */
 if (CreateThread( NULL, 0, comReadThread, (LPDWORD) rCom, NULL, 0 )==NULL)
 {
          printf("Fail to create a receiving thread\n");
          CloseHandle(wCom);
          CloseHandle(rCom);
          return 3;
 }
 for (i=0; i< MAX_DATA_LEN;i++)
          buffer[i] = (unsigned char) ('a'+i%26);

 PurgeComm(wCom,PURGE_TXCLEAR | PURGE_TXABORT);
 while(loop++ < 100)
 {
      if (WriteFile(wCom, buffer, MAX_DATA_LEN, &rtn, NULL)==0)
          {
                  printf("Fail to write\n");
                  break;
          }
          WaitForSingleObject(waitH, 100);
 }
 CloseHandle(wCom);
 CloseHandle(rCom);
 CloseHandle(waitH);

 return 0;
}

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
LPTSTR lpCmdLine, int nCmdShow)
{
 comPair(L"COM3", L"COM4", 38400);
 return 0;
}
```

Windows® API function *CreateFile(…)* opens a named file corresponding to a serial port. For any serial port from "COM10" or after, the file name must be prefixed by ***$device***\\. For "COM1" to "COM9", the ***$device***\\ prefix is optional. API function *GetCommState(..)* is the standard function to get the current parameters of a serial port, including its baudrate, data bits (bytesize), parity, and stop bits. In contrast to *GetCommState()*, API function *SetCommState(...)* controls serial communication by setting these parameters. Use other parameters including flow control, DTR signal, and RTS signal to fine tune the serial port. Be sure to set the flow control correctly or else you will lose data.

Depending on the serial port to which a device connects, the target computer supports serial communication in four operation modes: RS-232, RS-422, 2-wire RS-485, and 4-wire RS-485; RS-232 is the default. To change the operation mode, your program should include the following macro definitions; also insert the following *DeviceIoControl* function on the open handle before the program performs read/write operations.

```
/* --------- extracted from <moxa/devices> ------------------
#define MOXA_SET_OP_MODE    (0x400 + 66)
#define MOXA_GET_OP_MODE    (0x400+ 67)
#define RS232_MODE      0
#define RS485_2WIRE_MODE    1
#define RS422_MODE      2
#define RS485_4WIRE_MODE    3
*/

/* for example, set the mode to 4-wire RS485 */
BYTE mode = RS485_4WIRE_MODE;
DeviceIoControl(hCom,MOXA_SET_OP_MODE, &mode, sizeof(mode), NULL,0,NULL,NULL);
/* for example, get the current mode */
BYTE mode, size;
DeviceIoControl(hCom,MOXA_GET_OP_MODE, NULL, 0, (LPVOID)&mode,
sizeof(mode),&size,NULL);
```

Example #2—Buzzer

The target computer supports buzzer hardware for your applications to generate alarms on critical errors. You can set the frequency and the duration of the buzzer at the application level by using the APIs as follows. For example, the code shown below triggers the buzzer for 2000 milliseconds at frequency 500Hzs.

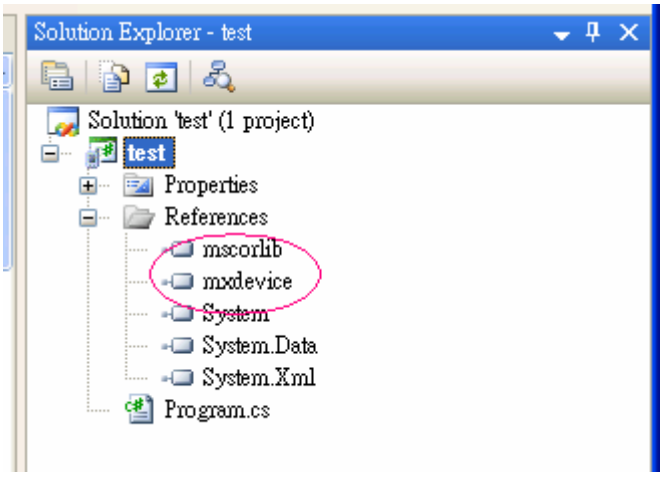
```
/* execute a beeper at a specified frequency for lasting a duration in miniseconds */
int mxbeep(unsigned int nHz, unsigned int nMiniSec);

#include <moxa/devices.h>

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
          LPTSTR    lpCmdLine, int      nCmdShow)
{
    mxbeep(500,2000);
   return 0;
}
```

Visual Studio 2005 C# Examples

To simplify application development with the Visual Studio 2005 tool, we provide you with a device .Net Compact Framework 2.0 class library (mxdevice.dll). To link the library with your project environment, perform the following steps from your Visual Studio 2005 environment:

- Copy the **mxdevice.dll** from the CD to your local disk. (CD path:\utility\NetCF20Lib\)
- Open the visual studio 2005 IDE tool, and then add a new **C# Smart device console application**.
- Input the project name and location path.

- In the **Solution Explorer View**, add **mxdevice.dll** to the reference section.
- Click **OK**



Example #1—MOXA UART Supports RS-232/422/485

The following C# code shows a sample application on operating a serial port. It provides methods to switch the operation mode of the port among RS-232, RS-422 and RS-485 modes. In addition, it uses an event based method to handle data reception.

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Data;
using System.Runtime.InteropServices;
using System.Collections.Specialized;
using System.IO.Ports;

namespace MxSerialPort
{
   class MxComPortConfig
    {
       [DllImport("coredll.dll", EntryPoint = "CreateFileW", SetLastError = true)]
       private static extern IntPtr CECreateFileW(string lpFileName, UInt32
dwDesiredAccess, UInt32 dwShareMode, IntPtr lpSecurityAttributes, UInt32
dwCreationDisposition, UInt32 dwFlagsAndAttributes, IntPtr hTemplateFile);

       [DllImport("coredll.dll", EntryPoint = "CloseHandle", SetLastError = true)]
       private static extern Int32 CECloseHandle(IntPtr hObject);

       [DllImport("coredll.dll", EntryPoint = "Sleep", SetLastError = true)]
       private static extern void CESleep(UInt32 dwMilliseconds);

       [DllImport("coredll.dll", EntryPoint = "DeviceIoControl", SetLastError = true)]
       public static extern bool DeviceIoControl(
          IntPtr hDevice,
          uint dwIoControlCode,
```

```
            ref byte lpInBuffer,
              int nInBufferSize,
            ref byte lpOutBuffer,
            int nOutBufferSize,
            ref int lpBytesReturned,
            IntPtr lpOverlapped
            );

        const uint MOXA_SET_OP_MODE = (0x400 + 66);
        const uint MOXA_GET_OP_MODE = (0x400 + 67);
        const uint RS232_MODE = 0;
        const uint RS485_2WIRE_MODE = 1;
        const uint RS422_MODE = 2;
        const uint RS485_4WIRE_MODE = 3;

        private readonly static IntPtr INVALID_HANDLE_VALUE = new IntPtr(-1);
        private const UInt32 OPEN_EXISTING = 3;
        private const UInt32 GENERIC_READ = 0x80000000;
        private const UInt32 GENERIC_WRITE = 0x40000000;

        String sPort;
        public MxComPortConfig(String sPort)
        {
            this.sPort = sPort;
        }

        private bool SetComPortInterface(uint mode)
        {
            IntPtr comPort = CECreateFileW(sPort, GENERIC_READ, 0, IntPtr.Zero,
OPEN_EXISTING, 0, IntPtr.Zero);
            if (comPort == INVALID_HANDLE_VALUE)
                return false;

            int nBytesReturned = 0;
            byte bIn = (byte)mode;
            byte bOut = 0;
            DeviceIoControl(comPort, MOXA_SET_OP_MODE, ref bIn, 1, ref bOut, 0, ref
nBytesReturned, IntPtr.Zero);
            CECloseHandle(comPort);
            return true;
        }

        public bool SetRS232()
        {
            return SetComPortInterface(RS232_MODE);
        }

        public bool SetRS422()
```

```
    {
        return SetComPortInterface(RS422_MODE);
    }

    public bool SetRS485TwoWire()
    {
          return SetComPortInterface(RS485_2WIRE_MODE);
    }

    public bool SetRS485FourWire()
    {
        return SetComPortInterface(RS485_4WIRE_MODE);
    }
}

class MxComPortEx
{
    private SerialPort port;

    private MxComPortEx(string sPortName, int baudrate)
    {
        // create a SerialPort instance with basic settings
        port = new SerialPort(sPortName, baudrate, Parity.None, 8, StopBits.One);

        port.RtsEnable = true;           // raise RTS for flow control
        port.ReadTimeout = 3000;         // 3 second timeout

        // Attach a method to be called when there is data waiting in the port's buffer
        port.DataReceived += new SerialDataReceivedEventHandler(DataReceived);
        port.ErrorReceived += new SerialErrorReceivedEventHandler(ErrorEvent);
    }

    private void Open()
    {
        if (!port.IsOpen)
            port.Open();
    }

    private void Close()
    {
        if (port.IsOpen) port.Close();
    }

    private void SendData(String str)
    {
        Console.WriteLine("Sending Data: " + str);
        port.WriteLine(str);
    }
```

```
        private void ErrorEvent(object sender, SerialErrorReceivedEventArgs e)
        {
            Console.WriteLine("error");
        }

        private void DataReceived(object sender, SerialDataReceivedEventArgs e)
        {
            while (port.BytesToRead > 0)
            {
                Console.WriteLine("Receiving Data: " + port.ReadByte());
            }
        }

        private bool IsOpen()
        {
            return port.IsOpen;
        }

        static void Main(string[] args)
        {
            MxComPortEx comPort = new MxComPortEx("COM3:", 115200);

            comPort.Open();
            if (comPort.IsOpen())
            {
                int count = 0;
                while (count++ < 10)
                {
                    comPort.SendData("ABCDEFGH" + count);
                }
                comPort.Close();
            }
        }
    }
}
```

Example #2—Buzzer

For example, the following code triggers the buzzer for 50 milliseconds at frequency 2200Hzs.

```
using System;
using System.Collections.Generic;
using System.Text;
using moxa;

namespace test
{
```

```
    class Program
    {
        static void Main(string[] args)
        {
            moxa.uc7400.Buzzer.Beep(2200, 50);
        }
}
}
```

Example #3—Real Time Clock

The following code shows you how to get the RTC Clock in system time structure format.

```
using System;
using System.Collections.Generic;
using System.Text;
using moxa;

namespace test
{
    class Program
    {
        static void Main(string[] args)
        {
            // get the RTC time
            moxa.mxdevice.SYSTEMTIME systime = moxa.uc7400.RTC.getRTC();
            System.Console.WriteLine("Year=" + systime.Year.ToString());
            System.Console.WriteLine("Month=" + systime.Month.ToString());
            System.Console.WriteLine("Day=" + systime.Day.ToString());
            System.Console.WriteLine("Hour=" + systime.Hour.ToString());
            System.Console.WriteLine("Minute=" + systime.Minute.ToString());
            System.Console.WriteLine("Second=" + systime.Second.ToString());
}
}
}
```

Frequently Asked Questions

Question	Answer	Steps
Can I set a connection port other than 80 for the web server (HTTPD)?	Yes. For example, your computer has a LAN port at 192.168.3.127 and a new HTTPD port 81 is chosen.	1. Open IE (Internet Explorer) to http://192.168.3.127/sysadmin/?Client=IE4 and logon as admin/admin. 2. Select the Registry Editor from the main menu. 3. Browse to **HKEY_LOCAL_MACHINE\Comm\HTTPD.**

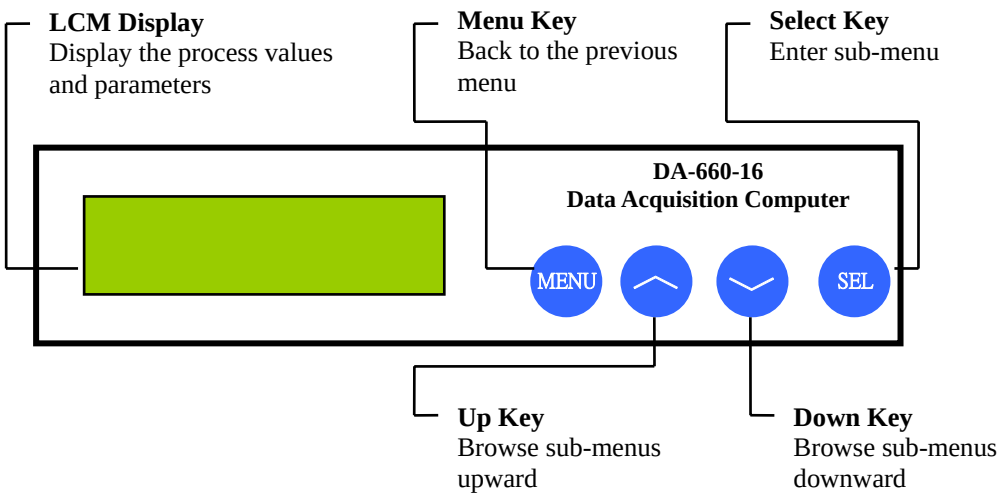
		4. Add a DWORD value named **Port**. (The default value must be zero). 5. Modify the value to **51** (the HEX value of **81**). 6. Browse to **HKEY_LOCAL_MACHINE\Services\HTTPD\Accept\TCP-80**. 7. Modify the value of **SockAddr** to "02 00 00 **51** 00 00 00 00 00 00 00 00 00 00 00 00". 8. Logon the computer via a telnet client and restart the web server. **/> services stop HTP0:** **/> services start HTP0:** 9. Try the server with the new URL http://192.168.3.127:81/
I have an eVB3.0 program. Can I run it on your computer?	No for Windows programs, the target computer is head-less without a screen. Applications running on it must be at the console level. To port your console program, just paste your source code into a new Visual Basic .Net 2005 console application project. There should be warnings, but you can correct it very easily with its dynamic help.	

I have a program under .NET Compact Framework 1.0 (CF1.0). Can I run it on your computer?	The target computer pre-installs an engine of version 2.0. It is compatible with version 1.0. Try to execute your program on the computer directly. If it does not work, open the project (CF1.0) with Visual Studio 2005 (CF2.0). You will see the migration messages and code warnings. Adjust the program step by step accordingly.	
I have a Windows Visual C++ program. Can I run it on your computer?	WinCE APIs are a subset of Win32 APIs. Care must be taken to migrate them. First, create a Visual Studio 2005 smart device VC project. Add all associated files to the project and rebuild the program. You might see compilation errors. Correct them with the help pages.	
I have a pure C program. Can I run it on your computer?	Create an eVC4 project or Visual Studio 2005 Smart Device VC project. Add your C files to your project, rewrite your main function as the winmain() function or the _tmain() function generated by the tool.	

A

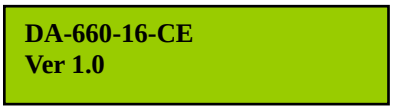
Using the LCM Keypad

Da-660 has an LCM and keypad that provide users with on-site parameter readings, and the ability to modify the configuration. Parameters that can be modified include model name, firmware version, network settings, etc. We use the DA-660-16-CE as an example to demonstrate the steps you should follow to configure these parameters.



Typical Example

Model Name and Firmware Version – This is the 1st screen when the system boots up.



Press **"MENU"** to enter the main menu.

Main Menu

Press **"﹀"** or use **"SEL"** to select an item.

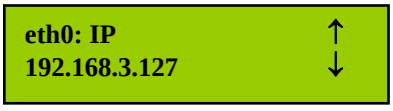
Press **"﹀"** or use **"SEL"** to select an item.

Press **"﹀"** or use **"SEL"** to select an item.

Network Settings (Port eth0 for example)

Press **"SEL"** .

Press **"SEL"** .



Press **"﹀"** .

Press **"﹀"** .

Press **"﹀"** .

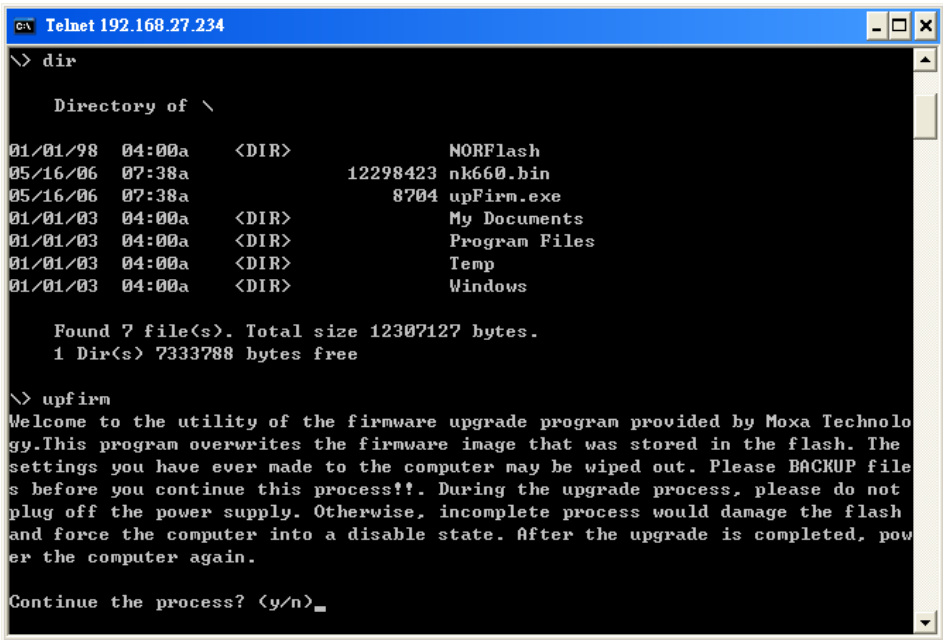
Press **"﹀"** .

B

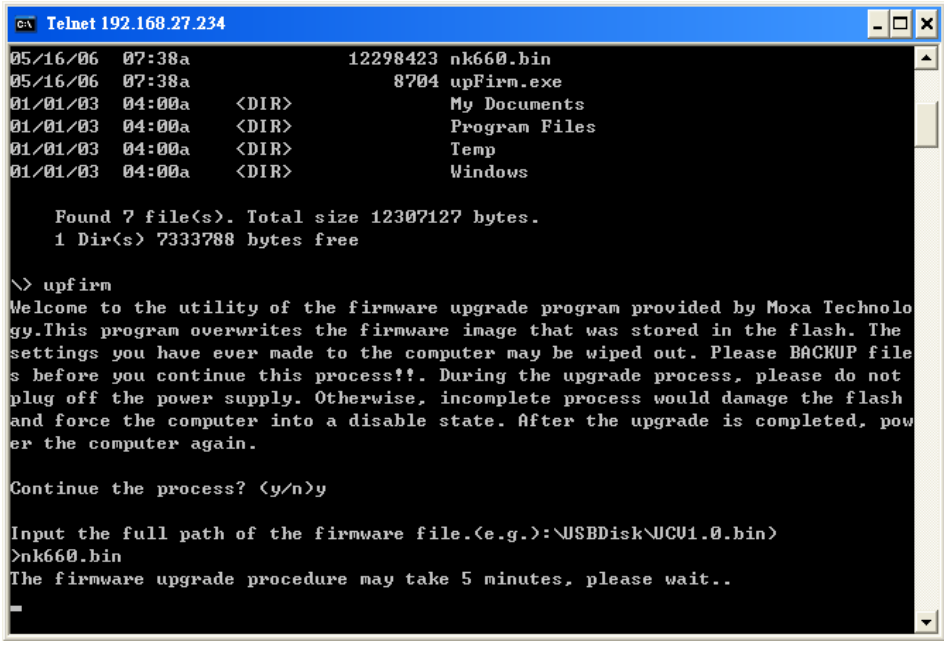
Firmware Upgrade Procedure

MOXA's engineering division continuously develops software features to improve the quality and functionality of MOXA's embedded products. The latest firmware is posted on the MOXA download center. To replace your product's firmware, perform the following steps:

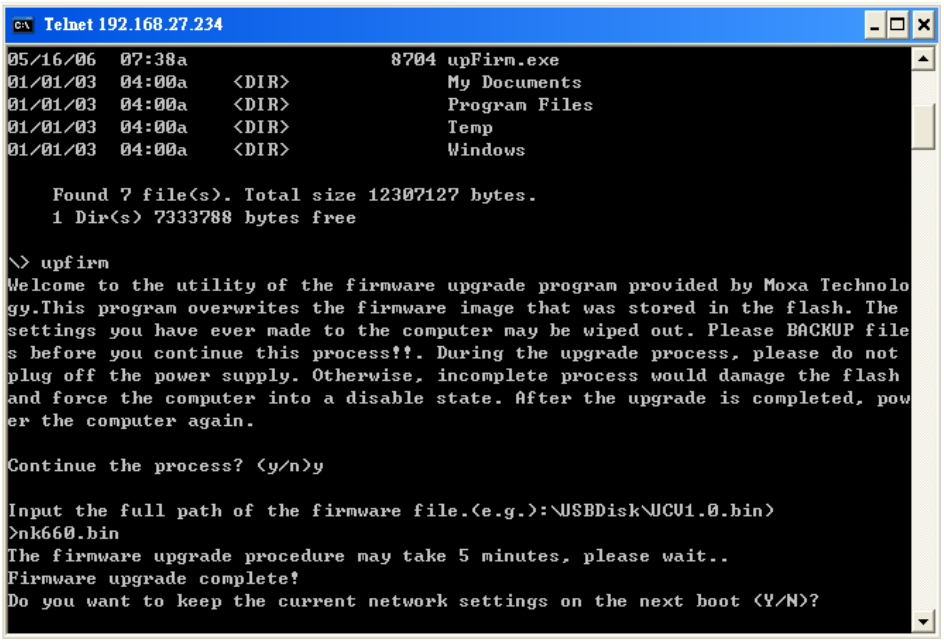
1. Navigate to the MOXA download center to download a program utility named ***upfirm*** and the firmware file (e.g., da660.bin) that you use to replace the current one.
2. Upload both files to the target machine under the root directory (i.e., \).
3. Logon the target computer via a telnet or console connection.
4. Execute the ***upfirm*** utility. Press **y** to continue.


Telnet 192.168.27.234
\> dir
Directory of \
01/01/98 04:00a <DIR> NORFlash
05/16/06 07:38a 12298423 nk660.bin
05/16/06 07:38a 8704 upFirm.exe
01/01/03 04:00a <DIR> My Documents
01/01/03 04:00a <DIR> Program Files
01/01/03 04:00a <DIR> Temp
01/01/03 04:00a <DIR> Windows
Found 7 file(s). Total size 12307127 bytes.
1 Dir(s) 7333788 bytes free
\> upfirm
Welcome to the utility of the firmware upgrade program provided by Moxa Technology.This program overwrites the firmware image that was stored in the flash. The settings you have ever made to the computer may be wiped out. Please BACKUP files before you continue this process!!. During the upgrade process, please do not plug off the power supply. Otherwise, incomplete process would damage the flash and force the computer into a disable state. After the upgrade is completed, power the computer again.
Continue the process? (y/n)

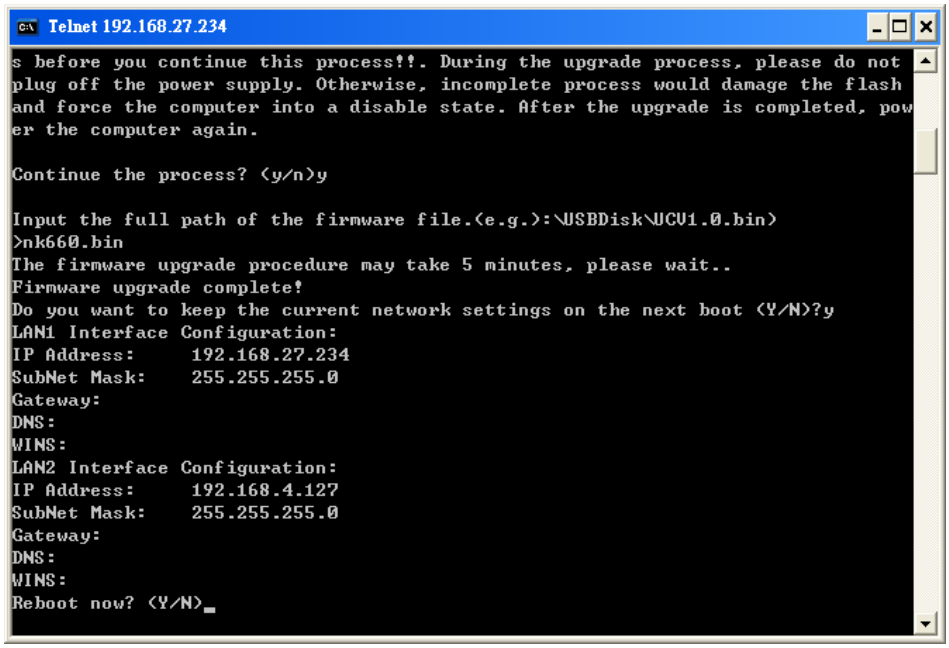
5. Input the full path of the firmware file and press Enter. The upgrade process takes about 5 minutes. The system should not be interrupted during this process.


Telnet 192.168.27.234
05/16/06 07:38a 12298423 nk660.bin
05/16/06 07:38a 8704 upFirm.exe
01/01/03 04:00a <DIR> My Documents
01/01/03 04:00a <DIR> Program Files
01/01/03 04:00a <DIR> Temp
01/01/03 04:00a <DIR> Windows
Found 7 file(s). Total size 12307127 bytes.
1 Dir(s) 7333788 bytes free
\> upfirm
Welcome to the utility of the firmware upgrade program provided by Moxa Technology.This program overwrites the firmware image that was stored in the flash. The settings you have ever made to the computer may be wiped out. Please BACKUP files before you continue this process!!. During the upgrade process, please do not plug off the power supply. Otherwise, incomplete process would damage the flash and force the computer into a disable state. After the upgrade is completed, power the computer again.
Continue the process? (y/n)y
Input the full path of the firmware file.(e.g.):\USBDisk\UCV1.0.bin)
>nk660.bin
The firmware upgrade procedure may take 5 minutes, please wait..

6. After the process is completed, enter "Y" to restore the current network settings or enter "N" to roll back to the factory default values.


Telnet 192.168.27.234
05/16/06 07:38a 8704 upFirm.exe
01/01/03 04:00a <DIR> My Documents
01/01/03 04:00a <DIR> Program Files
01/01/03 04:00a <DIR> Temp
01/01/03 04:00a <DIR> Windows
Found 7 file(s). Total size 12307127 bytes.
1 Dir(s) 7333788 bytes free
\> upfirm
Welcome to the utility of the firmware upgrade program provided by Moxa Technology.This program overwrites the firmware image that was stored in the flash. The settings you have ever made to the computer may be wiped out. Please BACKUP files before you continue this process!!. During the upgrade process, please do not plug off the power supply. Otherwise, incomplete process would damage the flash and force the computer into a disable state. After the upgrade is completed, power the computer again.
Continue the process? (y/n)y
Input the full path of the firmware file.(e.g.):\USBDisk\UCV1.0.bin)
>nk660.bin
The firmware upgrade procedure may take 5 minutes, please wait..
Firmware upgrade complete!
Do you want to keep the current network settings on the next boot (Y/N)?

7. Enter "Y" to reboot the target computer.


Telnet 192.168.27.234
s before you continue this process!!. During the upgrade process, please do not plug off the power supply. Otherwise, incomplete process would damage the flash and force the computer into a disable state. After the upgrade is completed, power the computer again.
Continue the process? (y/n)y
Input the full path of the firmware file.(e.g.):\USBDisk\UCV1.0.bin)
>nk660.bin
The firmware upgrade procedure may take 5 minutes, please wait..
Firmware upgrade complete!
Do you want to keep the current network settings on the next boot (Y/N)?y
LAN1 Interface Configuration:
IP Address: 192.168.27.234
SubNet Mask: 255.255.255.0
Gateway:
DNS:
WINS:
LAN2 Interface Configuration:
IP Address: 192.168.4.127
SubNet Mask: 255.255.255.0
Gateway:
DNS:
WINS:
Reboot now? (Y/N)

C

Service Information

This appendix shows you how to contact MOXA for information about this and other products, and how to report problems.

The following topics are covered in this appendix.

- **MOXA Internet Services**
- **Problem Report Form**
- **Product Return Procedure**

MOXA Internet Services

Customer satisfaction is our primary concern. To ensure that customers receive the full benefit of our products, MOXA Internet Services has been set up to provide technical support, driver updates, product information, and user's manual updates.

The following services are provided

E-mail for technical support...….....support@moxa.com

World Wide Web (WWW) Site for product information:

.....http://www.moxa.com

Problem Report Form

MOXA DA-660 Series

Customer name:	
Company:	
Tel:	**Fax:**
Email:	**Date:**

1. **Moxa Product:** ☐ DA-660-8-CE ☐ DA-660-16-CE
2. **Serial Number:** _____

Problem Description: Please describe the symptoms of the problem as clearly as possible, including any error messages you see. A clearly written description of the problem will allow us to reproduce the symptoms, and expedite the repair of your product.

Product Return Procedure

For product repair, exchange, or refund, the customer must:

- Provide evidence of original purchase.
- Obtain a Product Return Agreement (PRA) from the sales representative or dealer.
- Fill out the Problem Report Form (PRF). Include as much detail as possible for a shorter product repair time.
- Carefully pack the product in an anti-static package, and send it, pre-paid, to the dealer. The PRA should be visible on the outside of the package, and include a description of the problem, along with the return address and telephone number of a technical contact.